Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

# Функціонально-логічне проектування

## Лабораторний практикум

*Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського*
*як навчальний посібник для студентів,*
*які навчаються за спеціальністю 172 «Телекомунікації та радіотехніка»,*
*освітньої програми «Інформаційно-обчислювальні засоби електронних систем»*

Київ – 2019

Функціонально-логічне проектування. Лабораторний практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 172 «Телекомунікації та радіотехніка», освітньої програми «Інформаційно-обчислювальні засоби електронних систем» / КПІ ім. Ігоря Сікорського ; уклад.: Варфоломєєв А.Ю. – Київ : КПІ ім. Ігоря Сікорського, 2019. – 61 с.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 2 від 31.10.2019 р.)
за поданням вченої ради факультету електроніки (протокол № 10/2019 від 28.10.2019 р.)

Електронне навчальне видання

# Функціонально-логічне проектування
## Лабораторний практикум

Укладачі — *Варфоломєєв Антон Юрійович*, к.т.н., доц. каф. КЕОА

Відповідальний редактор — *Антонюк Олександр Ігорович*, ст. викл. каф. КЕОА

Рецензенти — Попов Антон Олександрович, к.т.н., доцент, доц. каф. електронної інженерії

Розглянуто практичні аспекти проектування цифрових комбінаційних пристроїв. Містяться короткі теоретичні відомості щодо основ алгебри логіки і теорії логічних функцій, способів представлення даних функцій у формах ДДНФ і ДКНФ, а також їх мінімізації за допомогою карт Карно. На основі викладеного теоретичного матеріалу наведено приклади синтезу найпоширеніших комбінаційних логічних пристроїв (дешифраторів, суматорів, мультиплексорів, схем зсуву та множення). Наведено методику проектування та реалізації даних пристроїв на ПЛІС (FPGA) із використанням середовища Intel Quartus та плат налагодження Terasic DE2.

Посібник може бути корисним для студентів та аспірантів радіотехнічних та телекомунікаційних спеціальностей, а також для інженерно-технічних спеціалістів, що працюють в галузі цифрової електроніки.

## ЗМІСТ

# ЛАБОРАТОРНА РОБОТА №1.
# Моделювання простих логічних схем на основі базових логічних елементів

**Тема роботи**: моделювання найпростіших логічних схем.

**Мета роботи**: ознайомитись з особливостями побудови та моделювання логічних схем в графічному редакторі Quartus II, навчитись виконувати моделювання логічних схем за допомогою вбудованого симулятора та із застосуванням плати налагодження.

## Основні теоретичні відомості

### Логічні елементи та базові логічні функції

*Логічний елемент І* (*AND*).

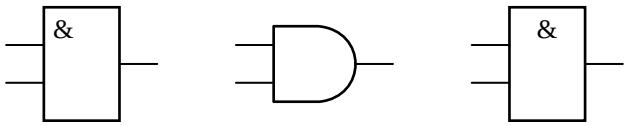
Логічний елемент І реалізує логічну операцію *кон'юнкції*. Словесний опис: на виході логічного елементу І формуватиметься *одиниця* тоді і тільки тоді, коли *на всі* його входи подано *одиниці*. Аналітичний запис операції кон'юнкції:

$$y = x_2 \cdot x_1 \text{ або } y = x_2 \wedge x_1 \text{ або } y = x_2 \,\&\, x_1$$

Таблиця істинності логічного елемента І з двома входами:

| $x_2$ | $x_1$ | $y = x_2 \wedge x_1$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Умовне графічне позначення:




ГОСТ ANSI IEC

*Логічний елемент АБО* (*OR*).

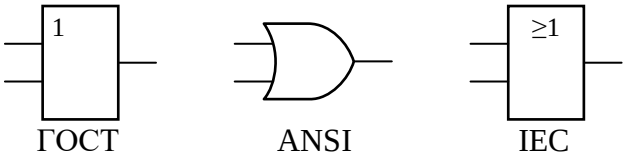
Логічний елемент АБО реалізує логічну операцію *диз'юнкції*. Словесний опис: на виході логічного елементу АБО формуватиметься *одиниця* тоді і тільки тоді, коли *хоча б на один* його вхід буде подано *одиницю*. Аналітичний запис операції диз'юнкції:

$$y = x_2 \vee x_1$$

Таблиця істинності логічного елементу АБО з двома входами:

| $x_2$ | $x_1$ | $y = x_2 \vee x_1$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

Умовне графічне позначення:




*Логічний елемент НІ* (*NOT*).

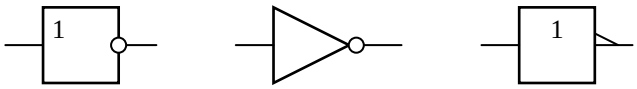
Логічний елемент НІ реалізує логічну операцію *заперечення*. Словесний опис: логічний елемент НІ формує на своєму виході сигнал протилежний тому, що подано на його вхід. Аналітичний запис операції заперечення:

$$y = \neg x \text{ або } y = \overline{x}.$$

Таблиця істинності для логічного елемента НІ:

| $x_1$ | $y = \overline{x}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

Умовне графічне позначення:




*Логічний елемент «Виключне АБО»* (*XOR*).

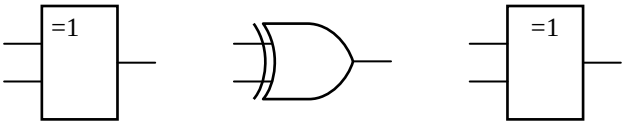
Логічний елемент «Виключне АБО» реалізує логічну операцію *сума по модулю* 2. Словесний опис функціонування: якщо елемент «Виключне АБО» має два вхідні сигнали, то на його виході формуватиметься *одиниця* тоді і тільки тоді, коли на обох його входах *сигнали відрізняються*. Аналітичний запис операції Виключне АБО:

$$y = x_2 \oplus x_1 = \overline{x}_2 \cdot x_1 \vee x_2 \cdot \overline{x}_1.$$

Таблиця істинності:

| $x_2$ | $x_1$ | $y = x_2 \oplus x_1$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Умовне графічне позначення:




ГОСТ ANSI IEC

**Теореми та тотожності алгебри логіки**

1) Ідемпотентні закони

$$\begin{cases} x \vee x = x \\ x \cdot x = x \end{cases}$$

2) Комутативні закони

$$\begin{cases} x \vee y = y \vee x \\ x \cdot y = y \cdot x \end{cases}$$

3) Асоціативні закони

$$\begin{cases} (x \vee y) \vee z = x \vee (y \vee z) \\ (x \cdot y) \cdot z = x \cdot (y \cdot z) \end{cases}$$

4) Дистрибутивні закони

$$\begin{cases} x \cdot (y \vee z) = x \cdot y \vee x \cdot z \\ x \vee y \cdot z = (x \vee y) \cdot (x \vee z) \end{cases}$$

5) Закони заперечення

$$\begin{cases} x \vee \bar{x} = 1 \\ x \cdot \bar{x} = 0 \end{cases}$$

$$\begin{cases} 0 \vee x = x \\ 1 \cdot x = x \end{cases}$$

$$\begin{cases} 1 \vee x = 1 \\ 0 \cdot x = 0 \end{cases}$$

6) Закони двоїстості (закони де Моргана)

$$\begin{cases} \overline{x \vee y} = \bar{x} \cdot \bar{y} \\ \overline{x \cdot y} = \bar{x} \vee \bar{y} \end{cases}$$

7) Закони подвійного заперечення

$$\overline{(\overline{x})} = \overline{\overline{x}} = x$$

8) Закони поглинання (абсорбції)

$$\begin{cases} x \vee x \cdot y = x \\ x(x \vee y) = x \end{cases}$$

9) Операції склеювання

$$\begin{cases} x \cdot y \vee x \cdot \bar{y} = x \\ (x \vee y) \cdot (x \vee \bar{y}) = x \end{cases}$$

10) Закони узагальненого склеювання

$$\begin{cases} x \cdot y \vee \bar{x} \cdot z \vee y \cdot z = x \cdot y \vee \bar{x} \cdot z \\ (x \vee y) \cdot (\bar{x} \vee z) \cdot (y \vee z) = (x \vee y) \cdot (\bar{x} \vee z) \end{cases}$$

$$\begin{cases} x \vee \bar{x} \cdot y = x \vee y \\ x \cdot (\bar{x} \vee y) = x \cdot y \end{cases}$$

**Пріоритетність операцій**

У алгебрі логіки, а отже і при записі логічних рівнянь прийнято наступну пріоритетність операцій: якщо в логічний вираз входять операції диз'юнкції та кон'юнкції, то кон'юнкція має вищий пріоритет виконання (виконується першою), а диз'юнкція – нижчий пріоритет. Порядок операцій можна довільно змінювати за допомогою дужок.

Якщо додатково в розгляд ввести операцію суми по модулю 2, то можна показати, що для того, щоб зберегти правильність результату виконання обчислень, її пріоритет має бути вищим за пріоритет диз'юнкції і нижчим за пріоритет кон'юнкції, тобто:

| Пріоритет *кон'юнкції* $(\cdot)$ | > | Пріоритет *суми по модулю* 2 $(\oplus)$ | > | Пріоритет *диз'юнкції* $(\vee)$ |
|---|---|---|---|---|

*Операція заперечення* є одномісною, тому *застосовується до аргументу в першу чергу*.

## Порядок виконання роботи

1. Побудувати в графічному редакторі системи Quartus II логічну схему відповідно до заданого варіанту (спрощення схеми не виконувати).
2. Створити файл з сигналами для моделювання, який би покривав всі можливі комбінації вхідних сигналів для схеми.
3. Провести моделювання у вбудованому симуляторі Quartus II та отримати часові діаграми. Впевнитись, що схема функціонує відповідно до заданого логічного рівняння.
4. Створити програму на зручній для Вас мові програмування, яка б реалізовувала роботу логічної схеми. Введення сигналів має бути реалізоване з клавіатури. Порівняти результати роботи програми з результатами моделювання в симуляторі Quartus II.
5. Обравши на платі налагодження необхідні засоби введення-виведення (кнопки та світлодіоди) модифікуйте схему та налаштуйте відповідні виводи ПЛІС. Створіть та завантажте прошивку на плату налагодження. Вводячи через кнопки чи перемикачі сигнали переконайтесь, що апаратна реалізація схеми функціонує згідно заданого логічного рівняння.

## Варіанти завдань

| | |
|---|---|
| 1. | $\overline{x_1 \cdot (x_1 \vee x_2)}$ |
| 2. | $\overline{\overline{x}_1 \cdot \overline{x_1 \vee x_2}}$ |
| 3. | $\overline{x}_1 \cdot \overline{x_1 \vee x_2}$ |
| 4. | $\overline{\overline{x}_1 \cdot (x_1 \vee \overline{x}_2)}$ |
| 5. | $\overline{x}_1 \cdot (x_1 \vee \overline{x}_2)$ |
| 6. | $\overline{x}_1 \cdot (x_1 \vee x_2)$ |
| 7. | $x_1 \cdot (\overline{x}_1 \vee \overline{x}_2)$ |
| 8. | $x_1 \cdot (\overline{x}_1 \vee x_2)$ |
| 9. | $x_1 \cdot \overline{\overline{x}_1 \vee x_2}$ |
| 10. | $x_1 \cdot \overline{\overline{x}_1 \vee \overline{x}_2}$ |
| 11. | $x_1 \vee \overline{\overline{x}_1 \cdot \overline{x}_2}$ |
| 12. | $x_1 \vee \overline{\overline{x}_1 \cdot x_2}$ |

| | |
|---|---|
| 13. | $\overline{x_1 \vee \overline{\overline{x}_1 \cdot \overline{x}_2}}$ |
| 14. | $\overline{x_1 \vee \overline{\overline{x}_1 \cdot x_2}}$ |
| 15. | $x_1 \vee \overline{x}_1 \cdot \overline{x}_2$ |
| 16. | $x_1 \vee \overline{x}_1 \cdot x_2$ |
| 17. | $\overline{x_1 \vee \overline{x}_1 \cdot \overline{x}_2}$ |
| 18. | $\overline{x_1 \vee \overline{x}_1 \cdot x_2}$ |
| 19. | $\overline{x}_1 \vee x_1 \cdot \overline{x}_2$ |
| 20. | $\overline{x}_1 \vee x_1 \cdot x_2$ |
| 21. | $\overline{\overline{x}_1 \vee x_1 \cdot \overline{x}_2}$ |
| 22. | $\overline{\overline{x}_1 \vee x_1 \cdot x_2}$ |
| 23. | $\overline{\overline{x}_1 \vee \overline{x_1 \cdot x_2}}$ |
| 24. | $\overline{\overline{x}_1 \vee \overline{x_1 \cdot \overline{x}_2}}$ |

## Контрольні питання

1. Які значення можуть приймати змінні в алгебрі логіки?
1. Які операції і відношення визначені в алгебрі логіки?
2. Скласти таблицю істинності для трьох входового логічного елемента І (входи: $x_3$, $x_2$, $x_1$, вихід: $y$).
3. Скласти таблицю істинності для трьох входового логічного елемента АБО (входи: $x_3$, $x_2$, $x_1$, вихід: $y$).
4. Скласти таблицю істинності для трьох входового логічного елемента І-НІ (входи: $x_3$, $x_2$, $x_1$, вихід: $y$).
5. Скласти таблицю істинності для трьох входового логічного елемента АБО-НІ (входи: $x_3$, $x_2$, $x_1$, вихід: $y$).
6. Скласти таблицю істинності для двох входового логічного елемента Виключне-АБО (входи: $x_2$, $x_1$, вихід: $y$).
7. Наведіть умовні графічні позначення логічних елементів в стандартах ГОСТ, ANSI, ISO/IEC.
8. Визначення системи числення. У чому відмінність позиційної системи числення від непозиційної? Навести приклади.
9. Закінчіть тотожності для операції «сума по модулю 2»:

$$x \oplus 0 = \ldots, \qquad x \oplus 1 = \ldots, \qquad x \oplus x = \ldots, \qquad x \oplus \overline{x} = \ldots$$

10. Наведіть пріоритетність виконання операцій в алгебрі логіки.
11. Спростіть логічний вираз власного та наступного варіантів (для останнього варіанта наступним вважати перший вираз). Назвіть логічну функцію, яку дані рівняння реалізують.

## ЛАБОРАТОРНА РОБОТА №2.

## Створення логічних схем на основі заданої таблиці істинності

**Тема роботи**: створення логічних схем за заданою таблицею істинності.

**Мета роботи**: ознайомитись зі способами створення логічних схем на основі заданої таблиці істинності шляхом побудови досконалих нормальних форм ДДНФ та ДКНФ.

### Основні теоретичні відомості

Закон функціонування будь-якого комбінаційного логічного пристрою (пристрою без пам'яті) може бути заданий *таблицею істинності*. Щоб за таблицю істинності побудувати логічний пристрій необхідно спершу отримати аналітичний запис функції, якій відповідає дана таблиця. Це робиться шляхом пошуку так званих *досконалої диз'юнктивної або досконалої кон'юнктивної нормальних форм*.

#### Досконала диз'юнктивна нормальна форма (ДДНФ)

*Досконала диз'юнктивна нормальна форма* (ДДНФ) є диз'юнкцією мінтермів, які відповідають таким наборам аргументів, на яких функція приймає лише одиничні значення.

*Мінтермом* (*конституентою одиниці*) називають функцію, яка приймає *одиничне* значення на одному з усіх можливих наборів аргументів та нульове значення при всіх інших наборах аргументів. Математичним виразом для мінтерма (або мінімального терма) є кон'юнкція *n* змінних або їх інверсій.

#### Досконала кон'юнктивна нормальна форма (ДКНФ)

*Досконала кон'юнктивна нормальна форма* (ДКНФ) є кон'юнкцією макстермів, які відповідають таким наборам аргументів, на яких функція, що розглядається приймає лише нульові значення.

*Макстермом* (*конституентою нуля*) називають функцію, яка приймає *нульове* значення на одному з усіх можливих наборів аргументів та одиничне

значення на всіх інших наборах аргументів. Математичним виразом для макстерма (або максимального терма) є диз'юнкція *n* змінних або їх інверсій.

Термін «досконала» у ДДНФ та ДКНФ означає, що всі члени розкладу мають однакову розмірність (однакову кількість змінних), а термін «нормальна форма» – що у виразах, які задають функцію, послідовно виконується *не більше двох* базових операції алгебри логіки без урахування операції заперечення.

**Приклад створення логічної схеми**

Нехай наведеною нижче таблицею істинності, задана деяка логічна функція, схему якої необхідно реалізувати.

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $y$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 | 1 |
| 5 | 0 | 1 | 0 | 1 | 1 |
| 6 | 0 | 1 | 1 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 |
| 10 | 1 | 0 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 | 1 | 0 |
| 12 | 1 | 1 | 0 | 0 | 1 |
| 13 | 1 | 1 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 | 1 |
| 15 | 1 | 1 | 1 | 1 | 0 |

Щоб реалізувати схему даної функції, її потрібно отримати в аналітичному вигляді. Це, як було зазначено вище, робиться за допомогою пошуку досконалих нормальних форм.

Застосуємо спершу метод побудови ДДНФ. Згідно визначення ДДНФ – це диз'юнкція мінтермів, які відповідають наборам аргументів, на яких функція приймає лише одиничні значення. Отже, у таблиці істинності

оберемо всі рядки, в яких функція дорівнює 1. Для наочності нижче наведемо таблицю істинності, в якій ці рядки відмічені.

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $y$ |
|---|---|---|---|---|---|
| 0 | **0** | **0** | **0** | **0** | **1** |
| 1 | **0** | **0** | **0** | **1** | **1** |
| 2 | 0 | 0 | 1 | 0 | 0 |
| 3 | **0** | **0** | **1** | **1** | **1** |
| 4 | **0** | **1** | **0** | **0** | **1** |
| 5 | **0** | **1** | **0** | **1** | **1** |
| 6 | 0 | 1 | 1 | 0 | 0 |
| 7 | **0** | **1** | **1** | **1** | **1** |
| 8 | 1 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 |
| 10 | **1** | **0** | **1** | **0** | **1** |
| 11 | 1 | 0 | 1 | 1 | 0 |
| 12 | **1** | **1** | **0** | **0** | **1** |
| 13 | **1** | **1** | **0** | **1** | **1** |
| 14 | **1** | **1** | **1** | **0** | **1** |
| 15 | 1 | 1 | 1 | 1 | 0 |

Тепер для кожного знайденого рядка необхідно записати мінтерми. За визначенням мінтерм – це кон’юнкція змінних або їх інверсій, яка приймає одиничне значення тільки на заданому наборі аргументів та нульове значення при всіх інших наборах аргументів. Таким чином, для рядка 0 маємо набір аргументів (0, 0, 0, 0), тоді єдина кон’юнкція змінних, яка прийматиме одиничне значення на такому наборі, очевидно, матиме вигляд $\bar{x}_4\bar{x}_3\bar{x}_2\bar{x}_1$. Аналогічно для рядка 1, на якому набір аргументів дорівнює (0, 0, 0, 1) мінтермом буде функція $\bar{x}_4\bar{x}_3\bar{x}_2x_1$. Для наступного рядка, у якому функція приймає одиничне значення – рядка 3 маємо набір аргументів (0, 0, 1, 1), для якого мінтерм матиме вигляд $\bar{x}_4\bar{x}_3x_2x_1$. Продовжуючи так далі знайдемо повний вираз для ДДНФ:

$$y = \bar{x}_4\bar{x}_3\bar{x}_2\bar{x}_1 \vee \bar{x}_4\bar{x}_3\bar{x}_2x_1 \vee \bar{x}_4\bar{x}_3x_2x_1 \vee \bar{x}_4x_3\bar{x}_2\bar{x}_1 \vee \bar{x}_4x_3\bar{x}_2x_1 \vee \\ \vee \bar{x}_4x_3x_2x_1 \vee x_4\bar{x}_3x_2\bar{x}_1 \vee x_4x_3\bar{x}_2\bar{x}_1 \vee x_4x_3\bar{x}_2x_1 \vee x_4x_3x_2\bar{x}_1 \qquad (2.1)$$

На основі знайденого виразу (2.1) тепер легко побудувати схему пристрою. Вона показана на рис. 2.1. Нижче на рис. 2.2 показано результат функціонального моделювання схеми, що реалізує ДДНФ.

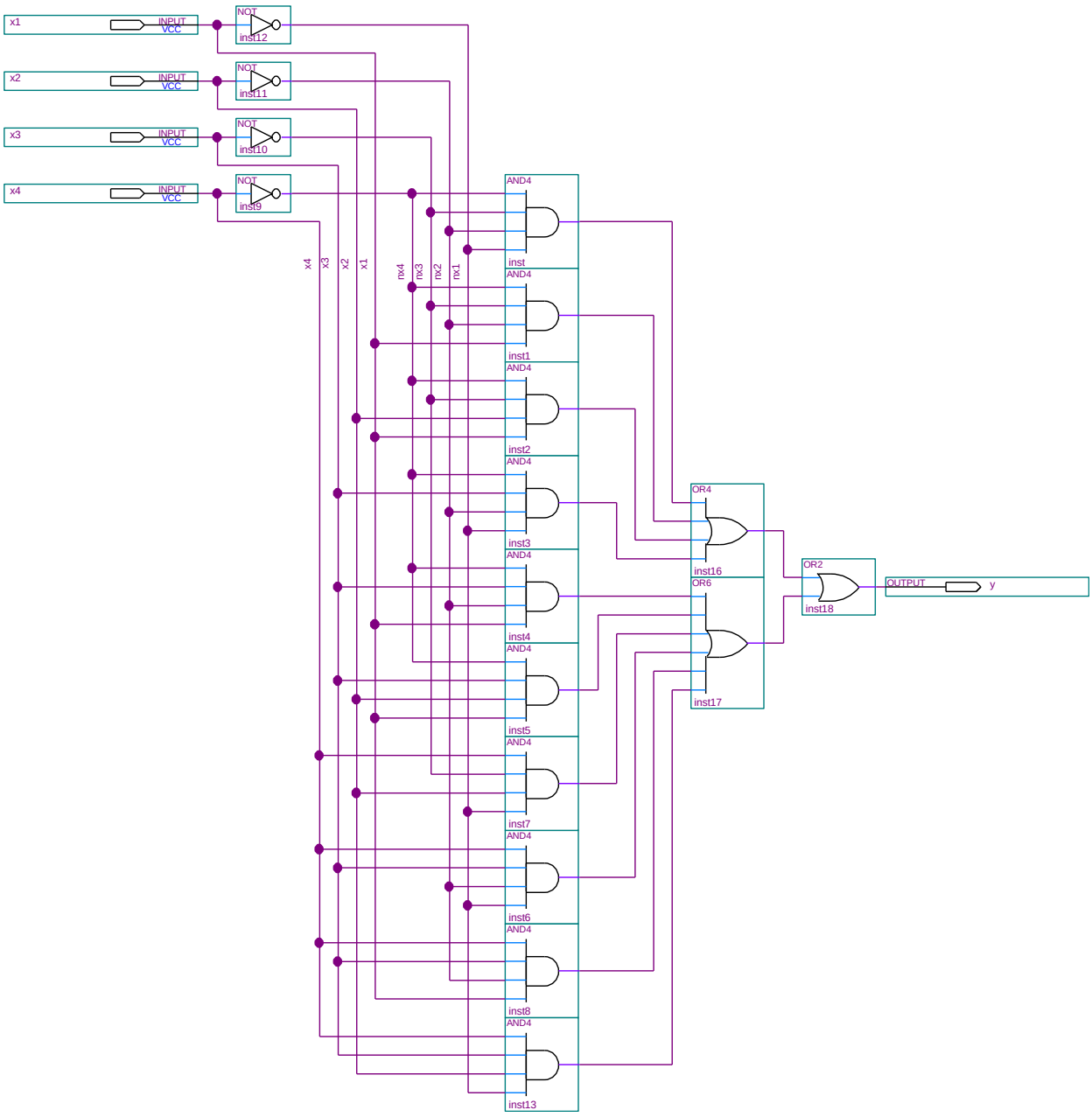



Рисунок 2.1 – Схема, що реалізує ДДНФ

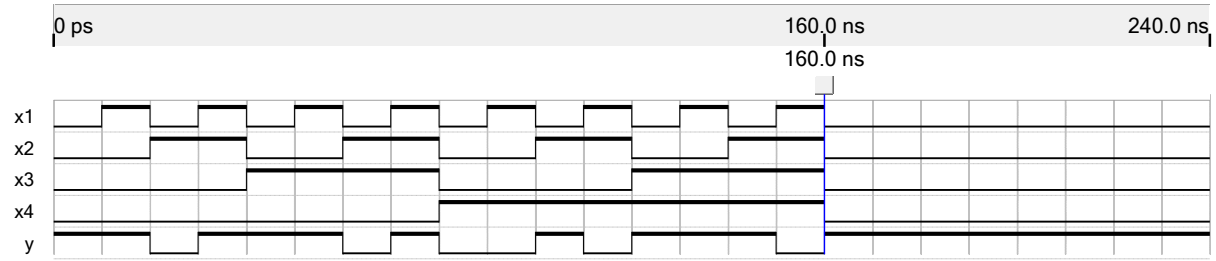



Рисунок 2.2 – Результат функціонального моделювання схеми, що реалізує ДДНФ

Згідно результатів моделювання можна бачити, що схема повністю реалізує поведінку, що задана таблицею істинності.

Тепер побудуємо ДКНФ функції. Згідно визначення ДКНФ – це кон'юнкція макстермів, які відповідають наборам аргументів, на яких функція приймає лише нульові значення. Отже, у таблиці істинності оберемо всі рядки, в яких функція дорівнює 0. Нижче наведено таблицю істинності, в якій ці рядки відмічені.

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $y$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 2 | **0** | **0** | **1** | **0** | **0** |
| 3 | 0 | 0 | 1 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 | 1 |
| 5 | 0 | 1 | 0 | 1 | 1 |
| 6 | **0** | **1** | **1** | **0** | **0** |
| 7 | 0 | 1 | 1 | 1 | 1 |
| 8 | **1** | **0** | **0** | **0** | **0** |
| 9 | **1** | **0** | **0** | **1** | **0** |
| 10 | 1 | 0 | 1 | 0 | 1 |
| 11 | **1** | **0** | **1** | **1** | **0** |
| 12 | 1 | 1 | 0 | 0 | 1 |
| 13 | 1 | 1 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 | 1 |
| 15 | **1** | **1** | **1** | **1** | **0** |

Для кожного знайденого рядка запишемо макстерми. За визначенням макстерм – це диз'юнкція змінних або їх інверсій, яка приймає нульове значення тільки на заданому наборі аргументів та одиничне значення при всіх інших наборах аргументів. Таким чином, для рядка 2 маємо набір аргументів (0, 0, 1, 0), а єдина диз'юнкція змінних, яка прийматиме нульове значення на такому наборі, очевидно, матиме вигляд $x_4 \vee x_3 \vee \bar{x}_2 \vee x_1$. Для рядка 6, на набір аргументів дорівнює (0, 1, 1, 0), тому макстермом буде функція $x_4 \vee \bar{x}_3 \vee \bar{x}_2 \vee x_1$. Продовжуючи так далі знайдемо повний вираз для ДКНФ:

$$y = (x_4 \vee x_3 \vee \bar{x}_2 \vee x_1) \cdot (x_4 \vee \bar{x}_3 \vee \bar{x}_2 \vee x_1) \cdot (\bar{x}_4 \vee x_3 \vee x_2 \vee x_1) \cdot \\ \cdot (\bar{x}_4 \vee x_3 \vee x_2 \vee \bar{x}_1) \cdot (\bar{x}_4 \vee x_3 \vee \bar{x}_2 \vee \bar{x}_1) \cdot (\bar{x}_4 \vee \bar{x}_3 \vee \bar{x}_2 \vee \bar{x}_1) \quad (2.2)$$

На основі знайденого виразу (2.2) також легко побудувати схему пристрою. Вона показана на рис. 2.3. Нижче на рис. 2.4 показано результат функціонального моделювання схеми, що реалізує ДКНФ.

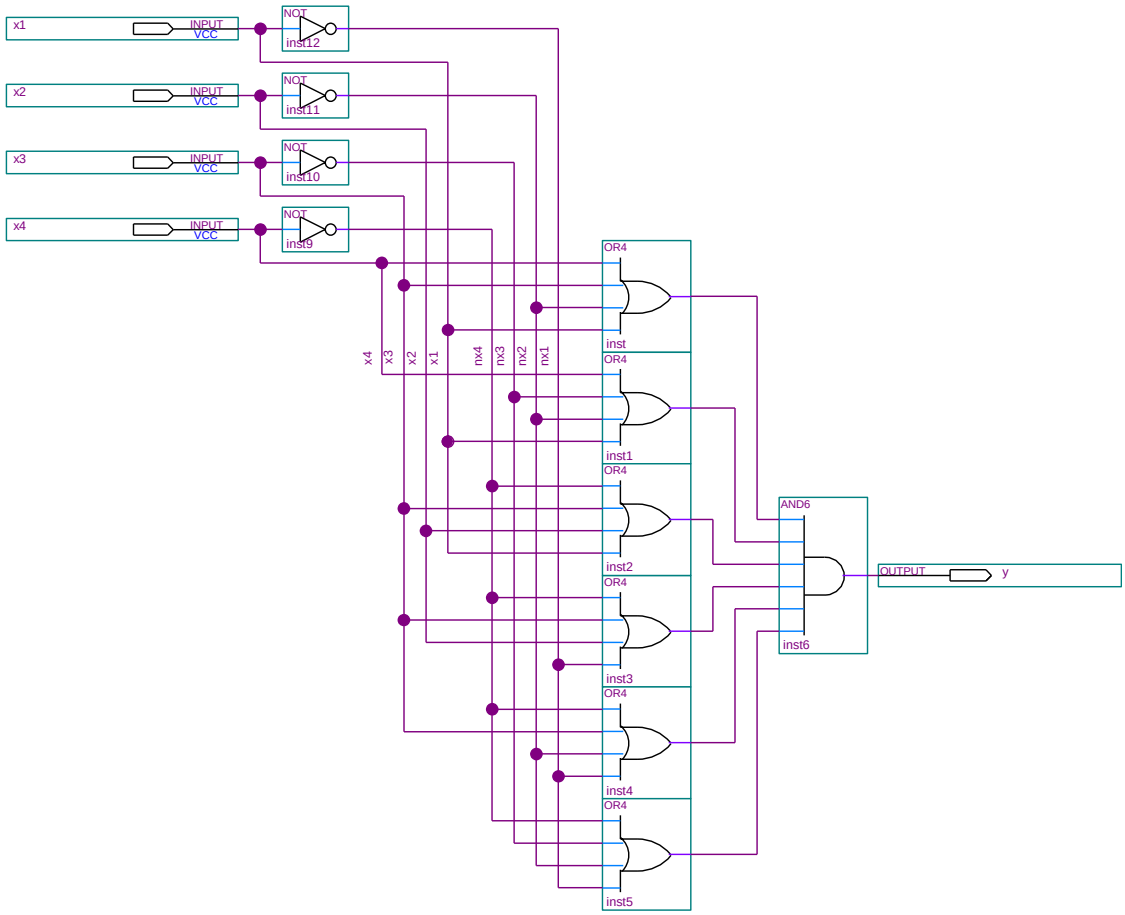



Рисунок 2.3 – Схема, що реалізує ДКНФ

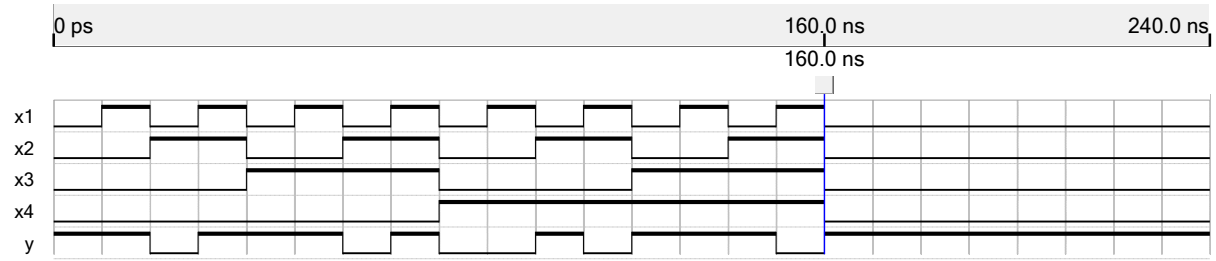



Рисунок 2.4 – Результат функціонального моделювання схеми, що реалізує ДКНФ

Як видно з результатів моделювання, схеми побудовані на основі ДДНФ та ДКНФ дають абсолютно однакові вихідні сигнали, що повністю відповідають заданій таблиці істинності. При цьому схема, отримана із ДКНФ є простішою, що обумовлено наявністю меншої кількості нулів, які має реалізовувати функція згідно таблиці істинності.

## Порядок виконання роботи

1. За таблицею істинності згідно заданого варіанту, отримати функцію алгебри логіки у вигляді:
    - досконалої диз’юнктивної нормальної форми (ДДНФ);
    - досконалої кон’юнктивної нормальної форми (ДДНФ);
    - ДДНФ інверсної функції;
    - ДКНФ інверсної функції;
2. Створити на основі отриманих ДДНФ та ДКНФ схеми логічних функцій у середовищі Quartus II.
3. Провести моделювання схем у вбудованому симуляторі Quartus II та отримати часові діаграми. Впевнитись, що схеми функціонують згідно таблиці істинності.
4. Перевірити правильність функціонування схеми згідно таблиці істинності використовуючи модуль налагодження. Виходи схем, отриманих на основі ДДНФ та ДКНФ, підключити до двох окремих діодів.

## Варіанти завдань

Варіант 1

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 2

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 3

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 4

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 5

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 6

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 7

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 8

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 9

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 10

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 11

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 12

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 13

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 14

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 15

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 16

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 17

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 18

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 19

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **1** |

Варіант 20

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 21

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **0** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 22

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **0** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **1** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **1** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 23

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **0** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **0** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **0** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **1** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Варіант 24

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $f(\nu_i)$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **0** |
| 3 | 0 | 0 | 1 | 1 | **0** |
| 4 | 0 | 1 | 0 | 0 | **0** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **0** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **0** |
| 10 | 1 | 0 | 1 | 0 | **1** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **1** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **1** |

## Контрольні питання

1. Наведіть закони двоїстості (закони де Моргана). В чому полягає суть принципу двоїстості?
2. Наведіть теорему розкладу Шеннона.
3. Виконайте розклад Шеннона функції за змінними $x_1$ та $x_2$:

$$f(x_4, x_3, x_2, x_1) = x_3x_1 \vee x_4x_2 \vee x_4x_1$$

4. Що таке таблиця істинності? Скільки рядків та стовпців має містити таблиця істинності і від чого це залежить?
5. Дайте визначення мінтерма. Наведіть приклад.
6. Дайте визначення макстерма. Наведіть приклад.
7. Дайте визначення досконалої диз'юнктивної нормальної форми (ДДНФ).
8. Дайте визначення досконалої кон'юнктивної нормальної форми (ДКНФ).
9. Що означає у ДДНФ та ДКНФ означають поняття «досконала» та «нормальна».
10. Реалізуйте логічні операції І та АБО двох змінних на основі операцій І-НІ та АБО-НІ.

## ЛАБОРАТОРНА РОБОТА №3.

## Оптимізація логічних схем.

## Синтез дешифраторів

**Тема роботи**: синтез дешифраторів та оптимізація логічних.

**Мета роботи**: навчитись синтезувати дешифратори та ознайомитись з методом мінімізації логічних функцій за допомогою карт Карно.

### Основні теоретичні відомості

*Дешифратор* (DC – decoder) – це логічний пристрій, який перетворює подане на його вхід двійкове число на сигнал необхідного рівня лише на одному з виходів даного пристрою, на всіх інших виходах, при цьому, формуються сигнали протилежного рівня.

*Повним дешифратором з прямими виходами* називається комбінаційна схема, що має $n$ входів та реалізує $2^n$ мінтермів:

$$F_i = K_i(\nu) = \prod_{p=1}^{n} x_p^{e_p} ,$$

де $\nu = (x_n,\ldots, x_1)$, $i = 0, 1,\ldots, 2^{n-1}$ – десяткове число, що еквівалентне двійковому числу $e_n,\ldots, e_1$. Згідно властивостей мінтермів при кожній комбінації вхідних сигналів $\nu$ тільки один вихід $F_i$ приймає значення рівне 1, тобто лише один вихід має високий рівень. Таким чином, повні дешифратори з прямими виходами виконують перетворення двійкового числа на унітарний код.

Позначення дешифраторів наведено на рис. 3.1.

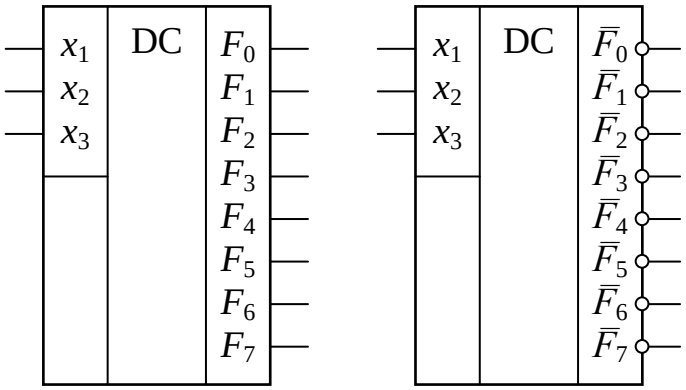



Рисунок 3.1 – Приклад позначення дешифраторів з прямими (зліва) та інверсними (справа) виходами

*Неповним дешифратором* називається комбінаційна схема, яка має $n$ входів та реалізує $N < 2^n$ мінтермів $n$ змінних.

*Дешифратор адреси* (ДшА) – це комбінаційна схема з $M$-входами та *одним* виходом, на якому формується сигнал високого рівня тоді, коли двійкове число, що подається на його входи, потрапляє в зазначений, при проектуванні дешифратора адреси діапазон.

Дешифратори адреси досить широко використовується в системах з *загальною шиною* – системах, в яких всі елементи одночасно підключені до єдиної шини даних та адреси. В даному випадку дешифратори адреси дозволяють визначити якому саме пристрою призначені дані: запит потрапляє до всіх елементів відразу, проте обробляється лише тим з них, чия адреса співпала з адресою, що містяться в запиті. Число входів $M$ дешифратора адреси при цьому залежить від кількості адресованих елементів, максимальне число яких дорівнює $2^M$.

Дешифратори адреси досить широко використовуються як один з функціональних блоків засобів введення-виведення.

**Приклад синтезу дешифратора адреси**

Необхідно синтезувати дешифратор 12-розрядної адреси, з діапазоном адрес 0xF00÷0xF0B, виключаючи адреси 0xF08 та 0xF0A.

Спершу синтезуємо перемикальну функцію, що описує роботу дешифратора адреси ($F$). Щоб спростити подальший синтез та не будувати таблицю істинності й карту Карно для 12-бітних функцій, в діапазоні адрес доцільно виділити дві частини – фіксовану, яка не змінюється в діапазоні та змінну. Нескладно бачити, що фіксованою частиною у даному випадку є 0xF0 (старші 8 біт). Зміна ж частина – це молодші 4 біти 0x0÷0xB, в які потрапляють також і виключення. Таким чином, логічну функцію, яку має реалізовувати дешифратор можна представити як:

$$F(x_{12},\dots x_1) = Y_{\text{фікс}}(x_{12},\dots, x_5) \cdot Y_{\text{змін}}(x_4,\dots, x_1),$$

де функція $Y_{\text{фікс}}$ – декодує старші 8 біт постійної частини адреси, а функція $Y_{\text{змін}}$ – молодші 4 біти, змінної частини адреси.

Спочатку отримаємо фіксовану частину адреси. Записувати таблицю істинності для функції, що реалізує цю частину не має сенсу, оскільки вона матиме 256 рядків, але тільки в одному з них вона дорівнюватиме 1. Цей рядок відповідає значенню 0xF0, або набору аргументів $(x_{12},\ldots,x_5) = 11110000_2$. Мінтерм, що відповідатиме даному набору аргументів матиме вигляд:

$$Y_{\text{фікс}} = x_{12}\cdot x_{11}\cdot x_{10}\cdot x_9\cdot \overline{x}_8\cdot \overline{x}_7\cdot \overline{x}_6\cdot \overline{x}_5$$

Тепер знайдемо функцію для змінної частини адреси. Цій частині, як було вже встановлено, відповідає 4 біти. Враховуючи це, а також наявні в точках 0x8 ($1000_2$) та 0xA ($1010_2$) виключення, запишемо таблицю істинності функції $Y_{\text{змін}}$:

| $i$ | $x_4$ | $x_3$ | $x_2$ | $x_1$ | $Y_{\text{змін}}$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | **1** |
| 1 | 0 | 0 | 0 | 1 | **1** |
| 2 | 0 | 0 | 1 | 0 | **1** |
| 3 | 0 | 0 | 1 | 1 | **1** |
| 4 | 0 | 1 | 0 | 0 | **1** |
| 5 | 0 | 1 | 0 | 1 | **1** |
| 6 | 0 | 1 | 1 | 0 | **1** |
| 7 | 0 | 1 | 1 | 1 | **1** |
| 8 | 1 | 0 | 0 | 0 | **0** |
| 9 | 1 | 0 | 0 | 1 | **1** |
| 10 | 1 | 0 | 1 | 0 | **0** |
| 11 | 1 | 0 | 1 | 1 | **1** |
| 12 | 1 | 1 | 0 | 0 | **0** |
| 13 | 1 | 1 | 0 | 1 | **0** |
| 14 | 1 | 1 | 1 | 0 | **0** |
| 15 | 1 | 1 | 1 | 1 | **0** |

Якщо за даною таблицею істинності безпосередньо побудувати ДДНФ, то вона може виявитись не оптимальною. Тому виконаємо мінімізацію функції $Y_{\text{змін}}$, для чого скористаємось картою Карно.

| $x_2x_1$ \ $x_4x_3$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 1 | 1 | 0 | 0 |
| 01 | 1 | 1 | 0 | 1 |
| 11 | 1 | 1 | 0 | 1 |
| 10 | 1 | 1 | 0 | 0 |

У карті Карно здійснюємо склеювання наступних рядків та стовпців: стовпці 00, 01 та всі рядки (відповідає терму $\bar{x}_4$), стовпці 00, 10 рядки 01, 11 (відповідають терму $\bar{x}_3 x_1$). В результаті отримуємо наступну МДНФ:

$$Y_{\text{змін}} = \bar{x}_4 \vee \bar{x}_3 x_1.$$

Для перевірки оберемо навмання декілька адрес та підставимо їх у триману функцію. Візьмемо, наприклад наступні адреси: 0x1, 0x8, 0x9, 0xA та 0xB, тоді:

$$0x1_{16}:\ 0001_2 \rightarrow \bar{0} \vee \bar{0} \cdot 1 = 1 \vee 1 \cdot 1 = 1$$
$$0x8_{16}:\ 1000_2 \rightarrow \bar{1} \vee \bar{0} \cdot 0 = 0 \vee 1 \cdot 0 = 0$$
$$0x8_{16}:\ 1001_2 \rightarrow \bar{1} \vee \bar{0} \cdot 1 = 0 \vee 1 \cdot 1 = 1$$
$$0xA_{16}:\ 1010_2 \rightarrow \bar{1} \vee \bar{0} \cdot 0 = 0 \vee 1 \cdot 0 = 0$$
$$0xB_{16}:\ 1011_2 \rightarrow \bar{1} \vee \bar{0} \cdot 1 = 0 \vee 1 \cdot 1 = 1$$

Схоже, що оптимальна функція для змінної частини адреси отримана вірно. Отже, запишемо остаточну функцію, що реалізує дешифратор адреси, об’єднуючи фіксовану та змінну чистини адреси:

$$F(x_{12},\ldots,x_1) = \underbrace{x_{12}x_{11}x_{10}x_9\bar{x}_8\bar{x}_7\bar{x}_6\bar{x}_5}_{Y_{\text{фікс}}} \cdot \underbrace{(\bar{x}_4 \vee \bar{x}_3 x_1)}_{Y_{\text{змін}}}$$

За отриманою функцією побудуємо схему дешифратора в середовищі Quartus II. Вона матиме вигляд, показаний на рис. 3.1.

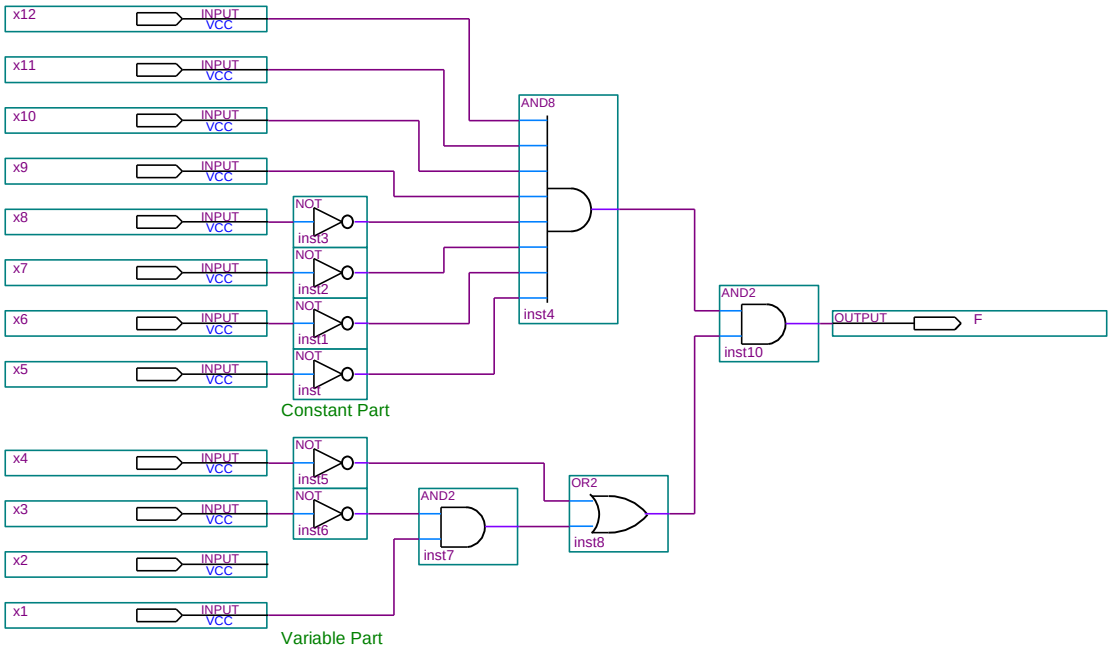



Рисунок 3.1 – Схема дешифратора адреси з діапазоном 0xF00÷0xF0B та виключеннями 0xF08 та 0xF0A

Як видно з функції дешифратора, в ньому не використовується змінна $x_2$. Тим не менш, на схемі відповідний вхід було залишено не підключеним. Це зроблено з одного боку для наочності, а з іншого – для більш зручного формування вхідних сигналів у файлі часових діаграм.

У редакторі часових діаграм, щоб не задавати окремо кожен вхідний сигнал доцільно згрупувати вхідні змінні. Це виконується наступним чином: слід виділити потрібні вхідні сигнали та в контекстному меню обрати Grouping… → Group (рис. 3.2).

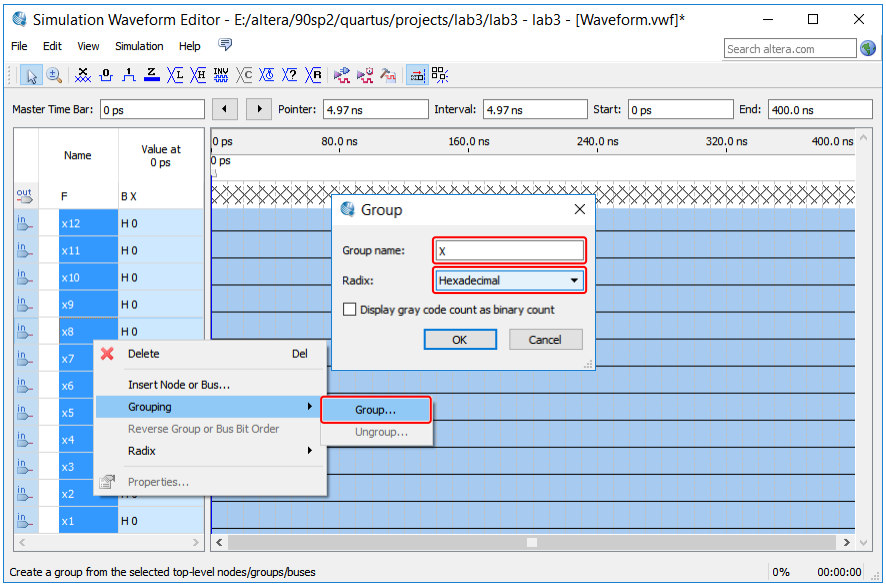



Рисунок 3.2 – Групування сигналів у редакторі часових діаграм

*Зауваження*. Після групування переконайтеся, що сигнали були впорядковані коректно – від старших до молодших розрядів. За необхідності перевпорядкуйте їх через контекстне меню Reverse Group or Bus Order.

Після створення групи сигналів їх можна задавати одразу у вигляді чисел, для чого натискати на кнопку . Сигнали також можна автоматично генерувати за допомогою спеціального засобу Count Value. Для цього необхідно виділити необхідний сигнал (або його частину) та натиснути на кнопку . У вікні, що відкриється слід задати початкове значення, з якого починається відлік (Start value), крок зміни (Increment by) та період зміни (Count every) див. рис. 3.3.

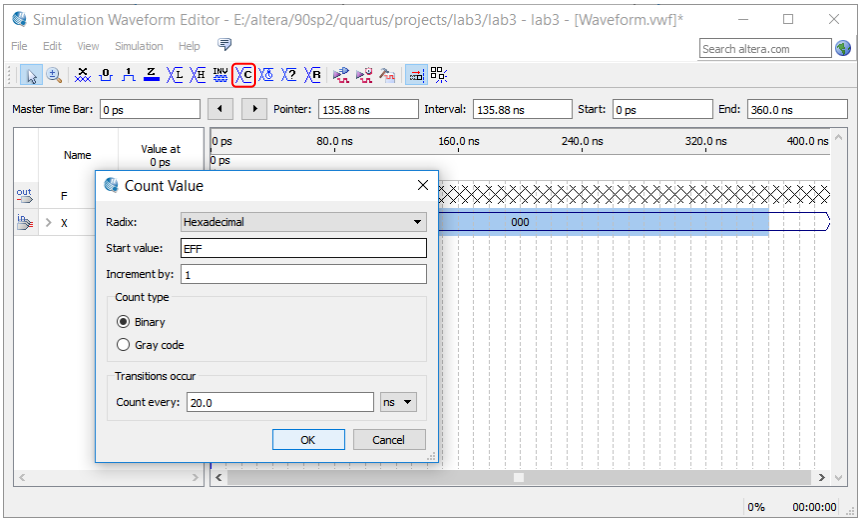



Рисунок 3.3 – Автоматичне генерування послідовності вхідних сигналів

Згенерувавши необхідні сигнали, виконуємо моделювання схеми. Результати моделювання показано на рисунку 3.4.

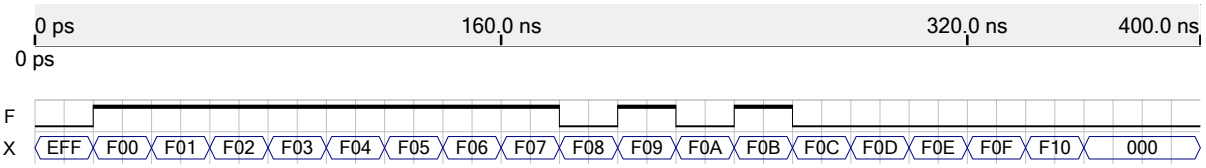



Рисунок 3.4 – Результати моделювання роботи дешифратора адреси

Отримані результати моделювання підтверджують, що дешифратор працює згідно заданого закону функціонування: він спрацьовує в діапазоні адрес від 0xF00 до 0xF0B та має два виключення у адресах 0xF08 та 0xF0A.

## Порядок виконання роботи

1. Згідно власного варіанта синтезувати оптимальну логічну функцію, що задає роботу дешифратора адреси.
2. У середовищі Quartus II за отриманою функцією побудувати схему дешифратора.
3. Провести моделювання схеми у вбудованому симуляторі Quartus II, отримати часові діаграми та впевнитись, що дешифрування адреси виконується вірно.
4. За допомогою налагоджувального модуля отримати апаратну реалізацію дешифратора адреси та продемонструвати його роботу.

5. Синтезувати дешифратор семисегментного індикатора, що виводить на індикатор шітнадцяткові цифри (для всіх варіантів). Типове позначення сегментів індикатора наведено у додатку Б (див. рис. Б.2) Провести моделювання та за допомогою модуля налагодження підтвердити працездатність отриманої схеми.

**Варіанти завдань**

| № | Початок діапазону | Кінець діапазону | Виключення |
|---|---|---|---|
| 1. | 0x201 | 0x207 | 0x203, 0x206 |
| 2. | 0x1F2 | 0x1FA | 0x1F6, 0x1F9 |
| 3. | 0x0E9 | 0x0EF | 0x0EA, 0x0EC |
| 4. | 0x502 | 0x50A | 0x505, 0x504 |
| 5. | 0x41C | 0x41F | 0x41D, 0x41E |
| 6. | 0x328 | 0x32C | 0x329, 0x32A |
| 7. | 0x230 | 0x23E | 0x23A, 0x23C |
| 8. | 0x143 | 0x14B | 0x145, 0x149 |
| 9. | 0x051 | 0x05A | 0x053, 0x054 |
| 10. | 0xF69 | 0xF6F | 0xF6A, 0xF6E |
| 11. | 0xE73 | 0xE7E | 0xE75, 0xE7A |
| 12. | 0xD81 | 0xD8D | 0xD82, 0xD84 |
| 13. | 0xC90 | 0xC9C | 0xC93, 0xC98 |
| 14. | 0xBA5 | 0xBAB | 0xBA8, 0xBAA |
| 15. | 0xAB4 | 0xABA | 0xAB8, 0xAB |
| 16. | 0x9C2 | 0x9C9 | 0x9C7, 0x9C8 |
| 17. | 0x8DA | 0x8DE | 0x8DB, 0x8DC |
| 18. | 0x7E4 | 0x7E8 | 0x7E6, 0x7E7 |
| 19. | 0x6F1 | 0x6F7 | 0x6F4, 0x6F5 |
| 20. | 0xF45 | 0xF4F | 0xF45, 0xF49 |
| 21. | 0xE54 | 0xE57 | 0xE55, 0xE56 |
| 22. | 0xD38 | 0xD3D | 0xD39, 0xD3A |
| 23. | 0xC60 | 0xC66 | 0xC63, 0xC65 |
| 24. | 0xB19 | 0xB1E | 0xB1A, 0xB1C |

## Контрольні питання

1. Що таке карта Карно? Які клітинки містять карти Карно, нумерація клітинок.
2. Якому закону булевої алгебри відповідає операція мінімізації за методом карт Карно?
3. Наведіть карту Карно для функції трьох змінних. Внесіть у клітинки цієї карти відповідні їм мінтерми.
4. Які клітинки карти Карно називаються сусідніми?
5. Які клітинки карти Карно можна об’єднувати в куби? Скільки клітинок може бути об’єднано в куби?
6. Сформулюйте задачу мінімізації функцій алгебри логіки за допомогою карт Карно.
7. Чи може одна клітинка карти Карно бути кубом?
8. Як складаються карти Карно для функцій з числом змінних більше, ніж 4? Які карти Карно є сусідніми?
9. В чому полягає особливість мінімізації неповністю визначених функцій? Наведіть приклад.
10. Наведіть процедуру мінімізації функцій алгебри логіки за допомогою метода Квайна-МакКласкі.
11. В чому полягають особливості спільної мінімізації декількох функцій.
12. Що таке дешифратор? Які дешифратори Вам відомі?
13. Що таке детектор стану?

# ЛАБОРАТОРНА РОБОТА №4.

## Пристрої двійкової арифметики

**Тема роботи**: створення основних пристроїв двійкової арифметики.

**Мета роботи**: навчитись синтезувати основні пристрої для виконання арифметичних дій над двійковими числами.

## Основні теоретичні відомості

### Суматори

Операція додавання будь-яких, в тому числі й двійкових чисел передбачає складання відповідних цифр $a_i + b_i$ доданків, причому якщо значення сум виявлятиметься більшим, ніж допустиме значення для цифри $a_i + b_i > p - 1$ в даній системі числення ( $p$ – основа системи числення), то має відбуватись перенесення *одиниці* у наступний розряд. Оскільки, пошук суми виконується порозрядно, то для реалізації $n$-розрядного суматора потрібно спочатку реалізувати схему, що правильно виконує додавання двох довільних розрядів числа, а потім на її основі можна буде створити повний $n$-розрядний суматор.

*Однорозрядний повний суматор*

Схема однорозрядного суматора повинна містити *три* входи: $a$ та $b$ для розрядів чисел, що додаються та вхід $c_i$ для врахування перенесення з попереднього розряду, а також *два* виходи: $s$ для виведення суми і вихід $c_o$ для перенесення у наступний розряд. Складемо таблицю істинності однорозрядного суматора.

| | $a$ | $b$ | $c_i$ | $s$ | $c_o$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 2 | 0 | 1 | 0 | 1 | 0 |
| 3 | 0 | 1 | 1 | 0 | 1 |
| 4 | 1 | 0 | 0 | 1 | 0 |
| 5 | 1 | 0 | 1 | 0 | 1 |
| 6 | 1 | 1 | 0 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 | 1 |

На основі отриманої таблиці істинності знайдемо ДДНФ функцій, що реалізують функції виходу $s$ та $c_o$:

$$s = \bar{a}\bar{b}c_i \vee \bar{a}b\bar{c}_i \vee a\bar{b}\bar{c}_i \vee abc_i = (\bar{a}\bar{b} \vee ab)c_i \vee (\bar{a}b \vee a\bar{b})\bar{c}_i =$$
$$= \overline{(a \oplus b)} \cdot c_i \vee (a \oplus b) \cdot \bar{c}_i = (a \oplus b) \oplus c_i;$$

$$c_o = \bar{a}bc_i \vee a\bar{b}c_i \vee ab\bar{c}_i \vee abc_i = (\bar{a}b \vee a\bar{b}) \cdot c_i \vee (\bar{c}_i \vee c_i) \cdot ab =$$
$$= (a \oplus b)c_i \vee ab.$$

Схема, що реалізує наведені вирази називається *повним суматором* та має вигляд, представлений на рис. 4.1.

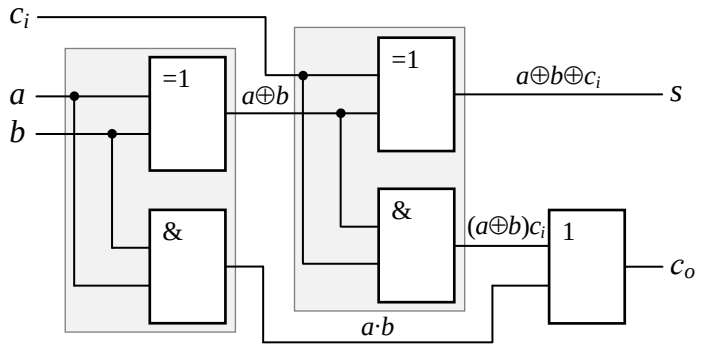



Рисунок 4.1 – Схема повного однобітного суматора

Вузли, що обведені в даній схемі пунктирною лінією називаються *напівсуматорами*. Повний суматор на схемах позначається як показано на рис. 4.2.

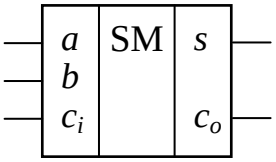



Рисунок 4.2 – Умовне позначення повного однобітного суматора

*Багаторозрядні суматори з послідовним перенесенням*

На основі схеми повного однобітного суматора з урахуванням перенесень можна будувати схеми суматорів з довільною розрядністю. Приклад схеми додавання 4-розрядних двійкових чисел, отриманої з 4 однобітних повних суматорів наведена нижче на рисунку 4.3.

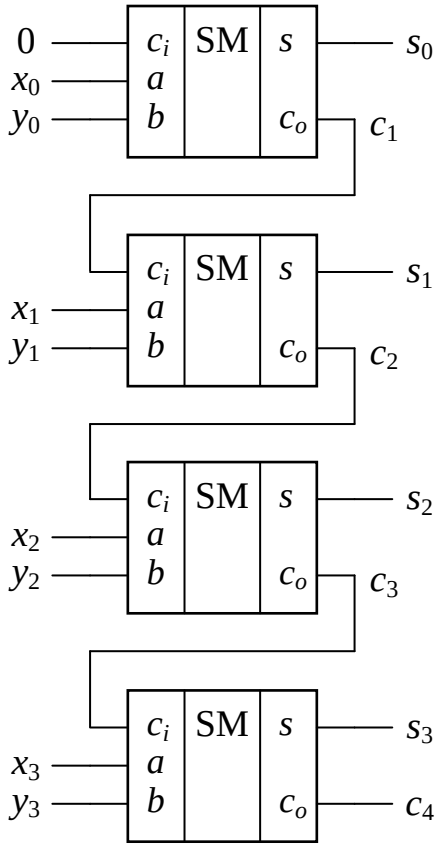



Рисунок 4.3 – Схема 4-розрядного суматора
з послідовним перенесенням

**Схема віднімання**

Операція віднімання може бути реалізована за допомогою переведення від'ємника у доповнювальний код та додавання його до зменшуваного. Перехід у доповнювальний код передбачає інвертування двійкового числа та додання до нього одиниці. З цього слідує, що для віднімання числа $B$ від числа $A$, необхідно до числа $A$ додати, інверсний код числа $\overline{B}$, а також одиницю. Побітове інвертування $B$ можна здійснити за допомогою логічних елементів НІ, а додавання одиниці – за рахунок подачі її на вхід перенесення нульового розряду суматора. Це дозволяє на базі багаторозрядного суматора синтезувати схему пристрою віднімання, показану на рис. 4.4.

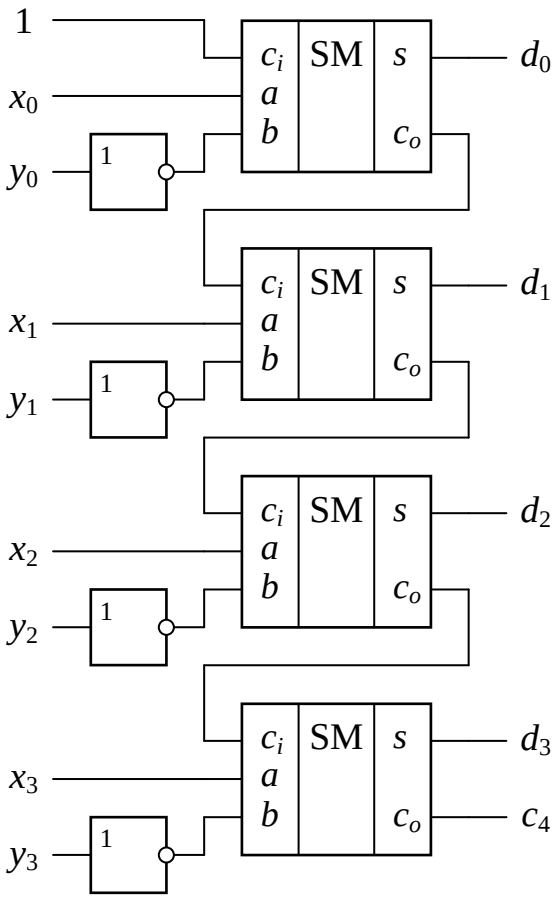



Рисунок 4.4 – Схема 4-розрядного пристрою віднімання, побудована на базі суматора з послідовним перенесенням

Для отримання універсальної схеми додавання/віднімання, в якій необхідна операція може обиратись, можна скористатись властивістю суми по модулю 2: двохвходовий логічний елемент Виключне АБО дозволяє інвертувати сигнал, що поступає на перший його вхід, якщо на другий вхід подано *одиницю* та пропускати сигнал без змін, при подачі на другий вхід *нуля*.

### Схеми комбінаційного зсуву

Схеми зсуву використовуються для зміщення розрядів двійкового числа на задану кількість позицій у заданому напрямку – вліво чи вправо. Операції зсуву, еквівалентні множенню (зсув вліво) та діленню (зсув вправо) даного числа на степінь двійки (основи системи числення). Існує декілька видів зсуву, що використовуються у цифрових пристроях, зокрема: *логічний*, *арифметичний* та *циклічний* (розглядалися на лекції).

Схема комбінаційного зсуву (Barrel Shifter) може бути побудована на основі мультиплексорів. Приклади схем логічного зсуву вліво та арифметичного зсуву вправо наведено на рис. 4.5.

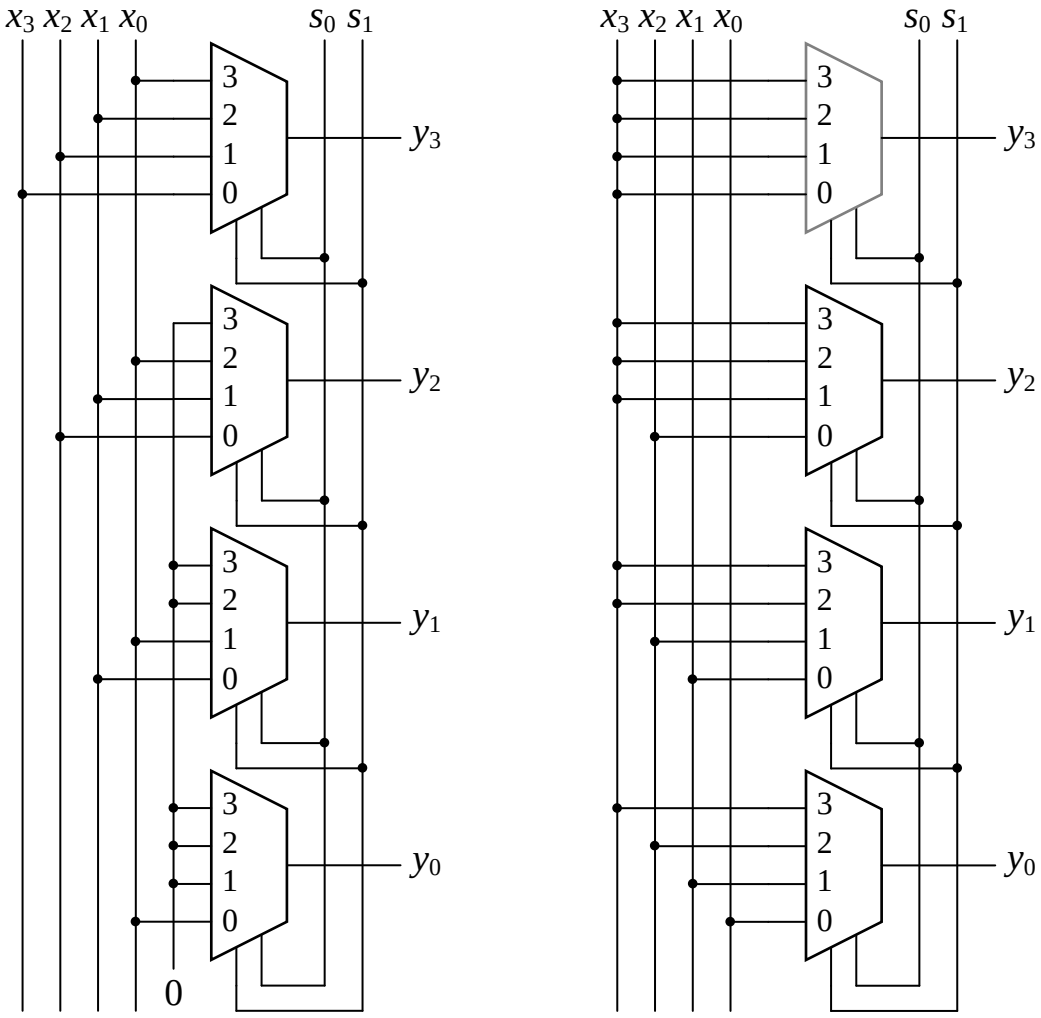



Рисунок 4.5 – Приклади комбінаційних 4-розрядних схем логічного та арифметичного зсуву

### Схема для виконання множення двійкових чисел

Множення двійкових чисел виконується за тими ж правилами, що й десяткове, за виключенням того, що дії здійснюються над нулями та одиницями. Базовою операцією, однорозрядного множення тут виступає операція *кон'юнкції*. При цьому виконується множення окремих розрядів множника на все множене, що дає частинні добутки. Далі, отримані частинні добутки зсуваються та додаються, даючи остаточний результат. Схематично

процедуру множення можна узагальнити наступною чином (для спрощення наведено 4-х розрядні числа):

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | $a_3$ | $a_2$ | $a_1$ | $a_0$ | множене |
| | | | | × $b_3$ | $b_2$ | $b_1$ | $b_0$ | множник |
| | | | | $a_3b_0$ | $a_2b_0$ | $a_1b_0$ | $a_0b_0$ | |
| + | | | $a_3b_1$ | $a_2b_1$ | $a_1b_1$ | $a_0b_1$ | | частинні добутки |
| | | $a_3b_2$ | $a_2b_2$ | $a_1b_2$ | $a_0b_2$ | | | |
| | $a_3b_3$ | $a_2b_3$ | $a_1b_3$ | $a_0b_3$ | | | | |
| $p_7$ | $p_6$ | $p_5$ | $p_4$ | $p_3$ | $p_2$ | $p_1$ | $p_0$ | результат |

Користуючись вже розробленим повним однобітним суматором, наведену вище процедуру можна представити комбінаційною схемою, представленою на рисунку 4.6.

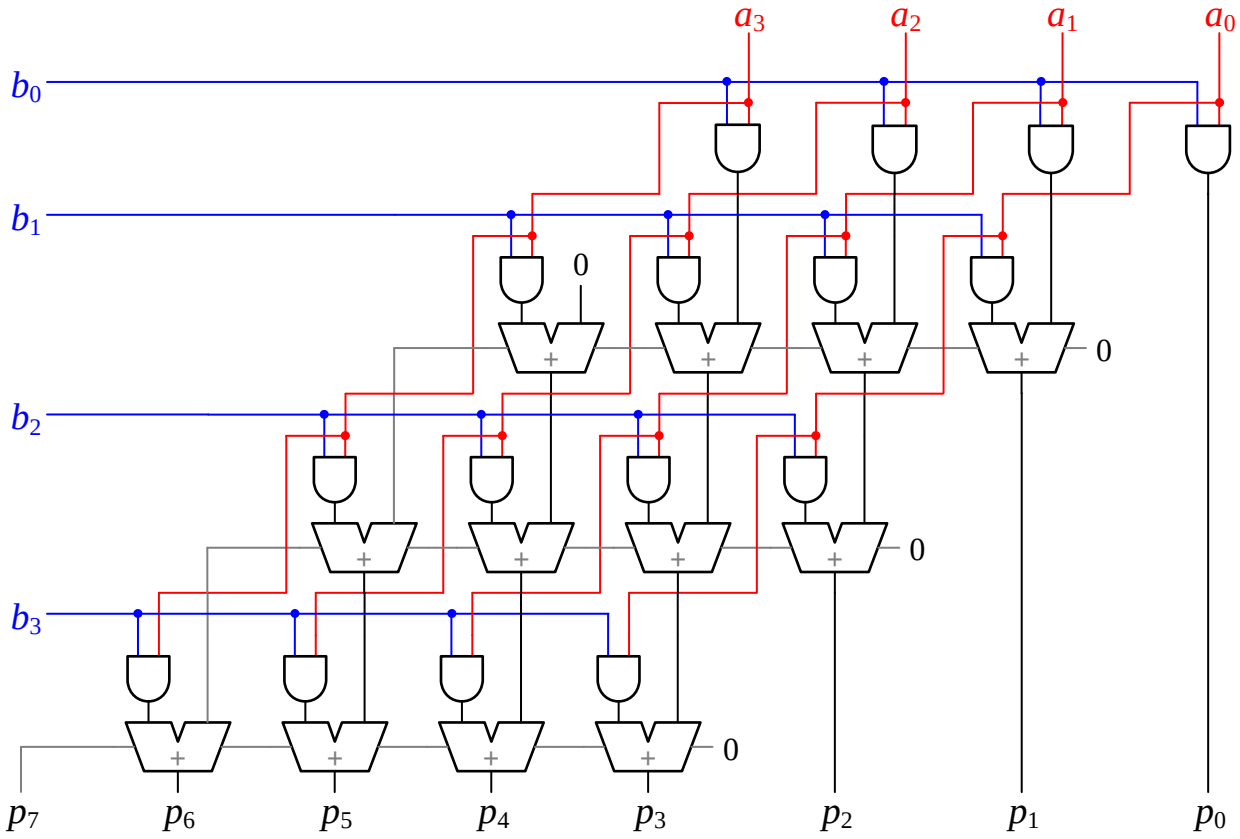



Рисунок 4.6 – Схема для виконання множення 4-х розрядних двійкових чисел

### Створення блоків модулів у середовищі Quartus II

Для більш зручного створення складних схем у середовищі Quartus II, елементарні їх складові, такі, наприклад, як однобітні суматори, мультиплексори та інші модулі, можна виносити в окремі блоки. Для цього необхідно створити окрему схему модуля (окремий *.bdf файл див. рис А.2),

в ньому за усіма правилами реалізувати схему потрібного пристрою (наприклад, однобітного повного суматора з рис. 4.1). Далі зайти в головне меню File → Create / Update → Create Symbol Files for Current File. Після цього буде запропоновано зберегти файл символу (*.bsf файл).

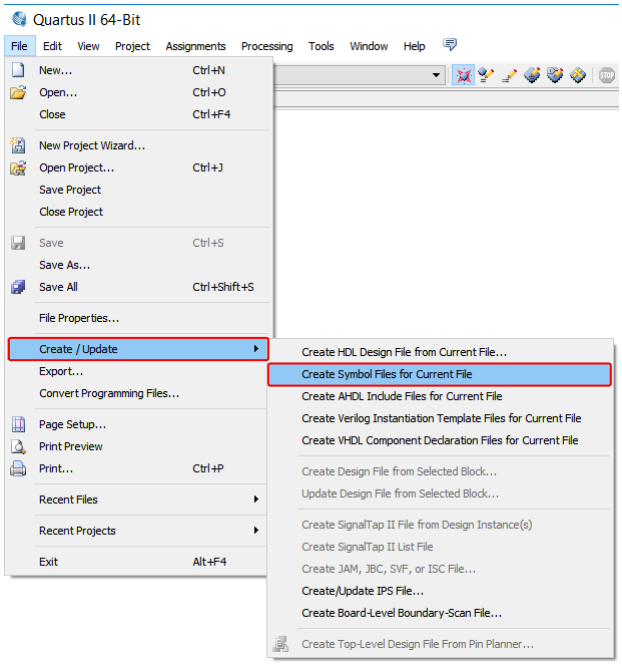



Рисунок 4.7 – Створення символа для модуля у середовищі Quartus II

Після того, як було створено символ, ним можна користуватися для створення інших схем у якості окремого блоку. При цьому користувацький символ (блок) має з'явитись у меню додавання компонентів на схему (рис. А.3, А.4), вкладці Project, як показано на рис. 4.8.

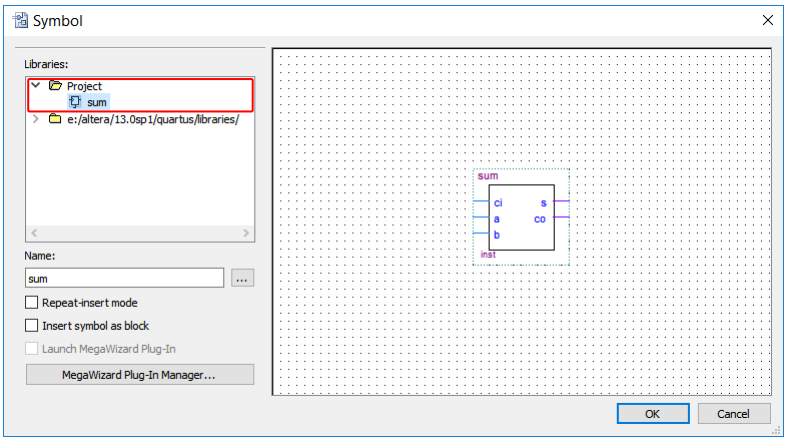



Рисунок 4.8 – Створений користувачем символ
у меню додавання компонентів

## Порядок виконання роботи

1. В середовищі Quartus II створіть базовий блок повного однобітного суматора, промоделюйте його та переконайтесь, що він працює коректно.
2. На основі повного однобітного суматора створіть універсальну схему, яка може виконувати додавання та віднімання 4-х розрядних двійкових чисел (підказка: для створення універсальної схеми необхідно замість інверторів у схемі з рис. 4.4 якимось чином використовувати елементи «Виключне АБО»). Реалізуйте дану схему на модулі налагодження. Для виведення результату скористайтеся створеним на минулій лабораторній роботі дешифратором семисегментного індикатора.
3. Створіть схему циклічного зсуву 4-х розрядних двійкових чисел, яка може виконувати зсув на задану кількість позицій $n$, $0 \le n < 4$ (студенти з парними номерами у списку реалізують схему зсуву вліво, а студенти з непарними номерами – вправо). Виведення результату виконайте у вигляді числа, що відображається на семисегментному індикаторі.
4. Створіть схему множення двох 3-бітових чисел. Результат множення виведіть на семисегментні індикатори модуля налагодження.

## Контрольні питання

1. Що таке доповнювальний код? Як до нього перейти зі звичайного (прямого) двійкового коду?
2. Виконайте арифметичні операції в двійковому вигляді, вважаючи що відповідні десяткові числа представляються 4-х розрядними двійковими значеннями:

$$\begin{array}{lll} 4 + 7 & 8 - 5 & 2 - 6 \\ -4 - 5 & 13 - 10 & 2 \times 4 \end{array}$$

3. На рисунку 4.1 наведено повний однобітний суматор. Чи можна створити більш оптимальну схему даного суматора?
4. Що таке напівсуматор, чим він відрізняється від повного суматора?
5. В чому полягають переваги і недоліки суматора з послідовним перенесенням в порівнянні з суматором з паралельним перенесенням?

6. Наведіть універсальну схема суматора, в якому за допомогою керуючого входу можна обирати дію додавання та віднімання.
7. Назвіть існуючі види схем зсуву та поясніть їх особливості.
8. В чому полягає особливість арифметичного зсуву, для чого це необхідно?
9. Наведіть алгоритм множення двійкових чисел.
10. Наведіть алгоритм ділення двійкових чисел.

# ДОДАТОК А
# ОСНОВИ РОБОТИ В СИСТЕМІ QUARTUS II ТА ЗАСТОСУВАННЯ ПЛАТ НАЛАГОДЖЕННЯ

## Створення проекту

Для створення проекту в середовищі розробки Quartus II необхідно зайти в головне меню File → New Project Wizard, після чого з'явиться майстер створення проектів показаний на рис. А.1.

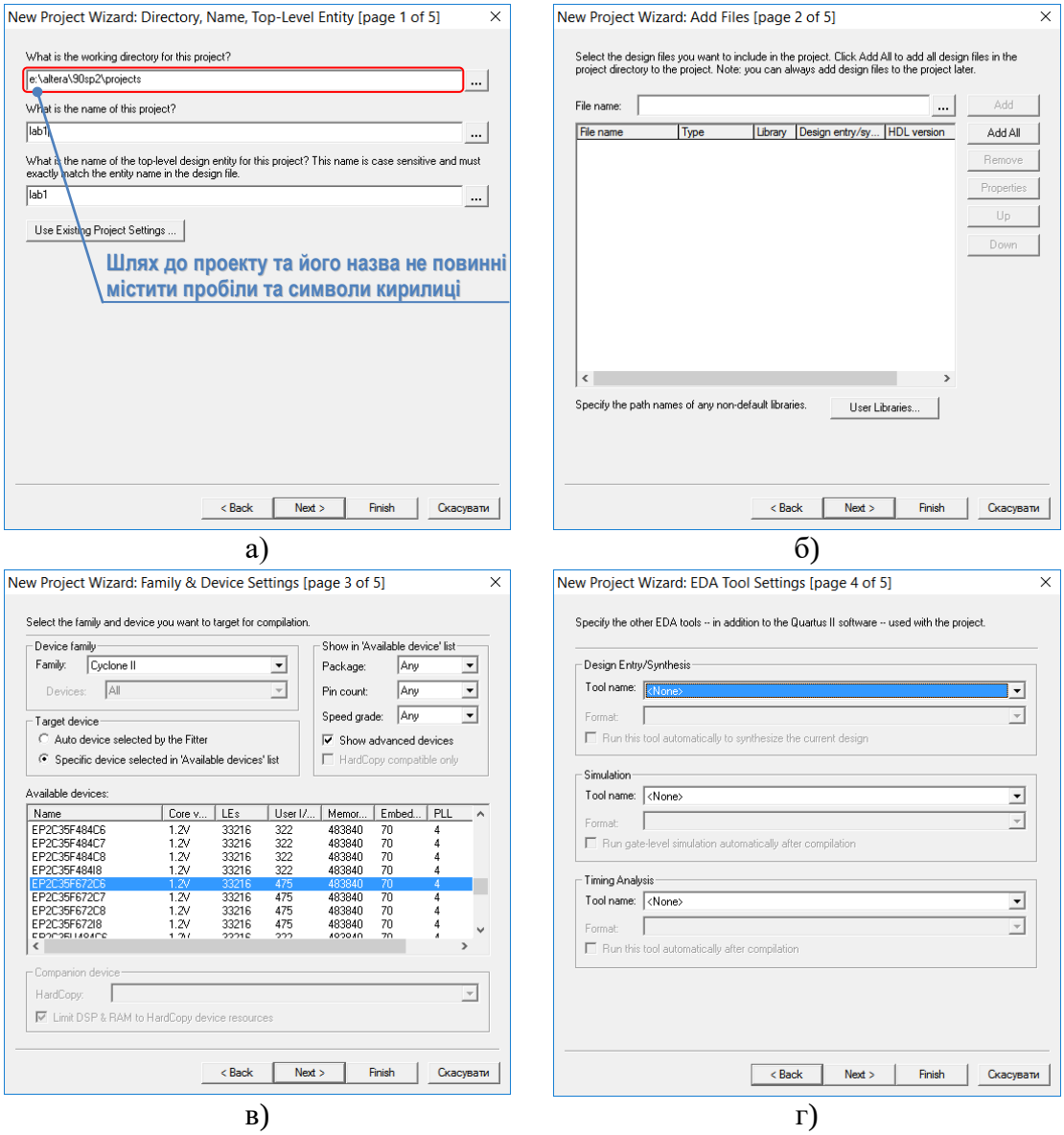



Рисунок А.1 – Майстер створення проектів:

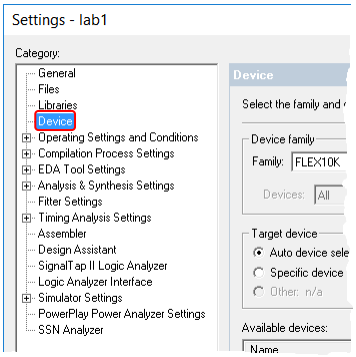
а) у даному вікні задається шлях розміщення проекту та його ім'я;

б) додавання існуючих компонентів, модулів та бібліотек;

в) вибір додаткових засобів моделювання та аналізу;

г) вибір сімейства ПЛІС, для якого розробляється проект

На сторінці 3 (рис. А.1, *в*) вибір ПЛІС виконується відповідно до мікросхеми, яка встановлена на платі налагодження. Зокрема, на платі DE2 використовується мікросхема сімейства Cyclon II (EP2C35F672C6), а на платі UP2 наявні дві мікросхеми сімейств Flex (EPF10K70RC240-4) та Max 7000 (EPM7128SLC84-7). Плата UP2 сконфігурована так, що за замовчуванням використовується ПЛІС Flex.




Якщо сімейство ПЛІС не було встановлено вірно відразу при створенні проекту, його можна змінити пізніше у налаштуваннях: меню Assigments → Device або Assigments → Settings вкладка Device.

## Створення схеми логічного пристрою

Для створення схеми необхідно зайти в меню File → New та у вікні, що відкриється та обрати Block Diagram/Schematic File (рис. А.2).

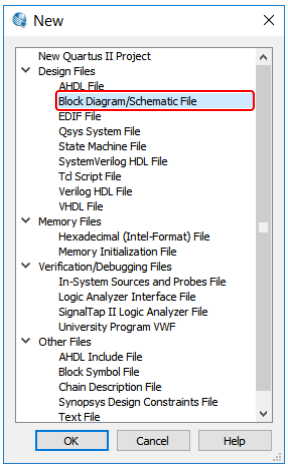



Рисунок А.2 – Створення схеми

Створення схеми виконується в спеціальному редакторі. Вікно даного редактора показано на рисунку А.3.

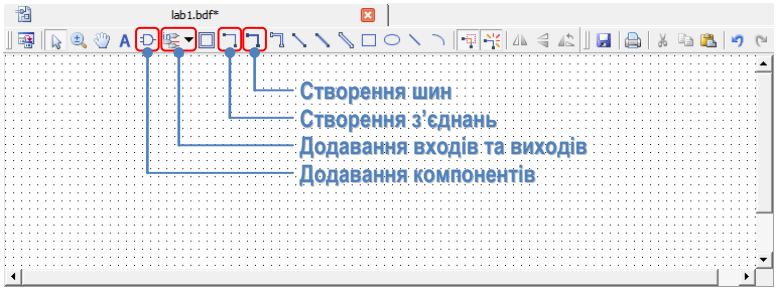



Рисунок А.3 – Редактор схем

Додавання компонентів до схеми виконується за допомогою спеціального інструменту, показаного на рис. А.4. Щоб його викликати необхідно натиснути на відповідну кнопку редактора схем (рис. А.3).

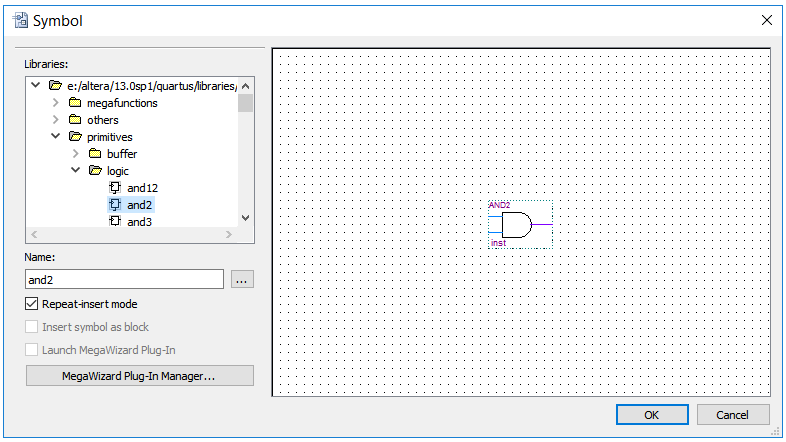



Рисунок А.4 – Інструмент додання компонентів

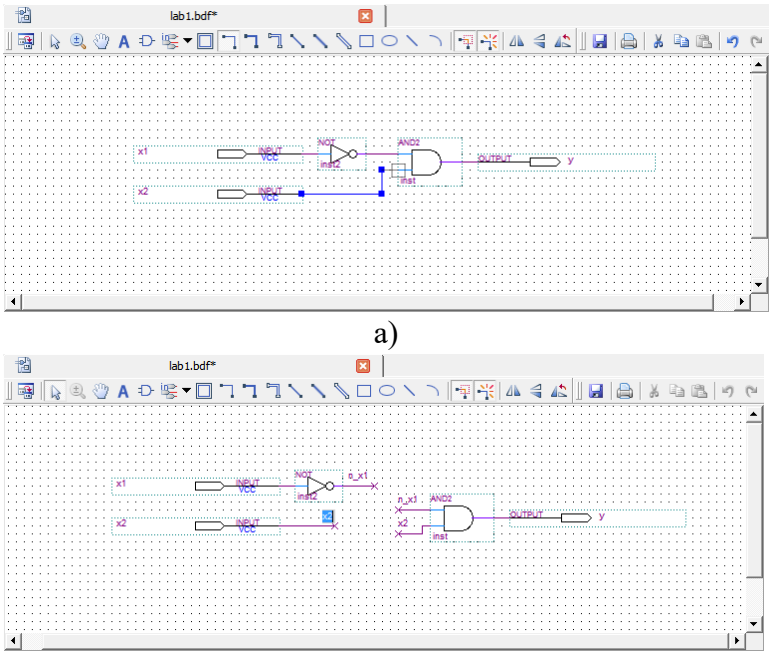
Вхідні та вихідні сигнали у схему можна додати за допомогою окремого інструменту доступного в редакторі (рис. А.3). Цей інструмент відсутній у старих версіях Quartus, тому там входи та виводи додаються як звичайні компоненти – за допомогою майстра наведеного на рис. А.4. При використанні майстра компонентів входи і виходи знаходяться у гілці primitives → pin. За необхідності імена входів та виходів можна змінювати. Для цього необхідно зайти у властивості (Properties) через контекстне меню або зробити подвійний клік на компоненті входу/виходу. Для зручності та уникнення помилок входам та виходам рекомендується давати такі ж імена, як для змінних, що використовуються у логічних функціях, на основі яких створюється схема.

Після розміщення всіх необхідних на схемі компонентів між ними мають бути створені з'єднання рис. А.5, *а*. Це можна здійснити декількома способами. Найпростішим з них є використання інструменту з'єднань (рис. А.3), який дозволяє поєднувати між собою входи та виходи компонентів за допомогою провідника (див. рис. А.5, *а*).

Якщо з'єднання є однотипними або багаторозрядними сигналами, то в цьому випадку доцільно утворювати шини. Це дозволяє зробити схему більш

компактною та зручною для читання. При створенні шин окремі провідники, що в них входять/виходять мають бути підписані. Для створення підпису, провідник необхідно просто виділити та ввести з клавіатури його ім’я. Також надати ім’я провіднику можна через контекстне меню, що викликається правою кнопкою миші. У контекстному меню слід увійти до налаштувань (Properties) та у вікні, що відкриється вести ім’я в поле Name.

Варто відмітити, що іменування провідників взагалі дозволяє не утворювати повних з’єднань, як показано на рис. А.5, *б*.




а)

б)

Рисунок А.5 – Утворення з’єднань у схемі:
а) за допомогою інструменту Orthogonal Node Tool;
б) за допомогою іменування провідників

Для перевірки правильності побудови схеми необхідно виконати «аналіз та синтез» проекту або здійснити його компілювання. Відповідні операції можна запустити через вкладку головного меню Processing → Start Compilation (комбінація клавіш Ctrl + L) та Processing → Start → Start Analysis & Synthesis (комбінація клавіш Ctrl + K) або натисканням кнопок показаних на рис. А.6.

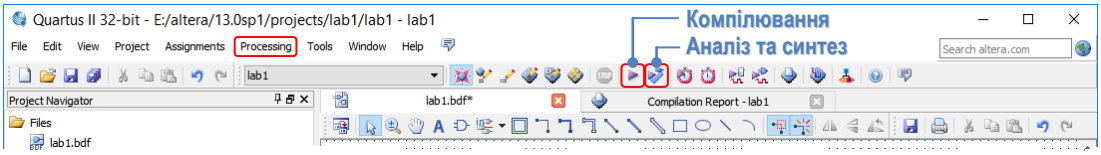



Рисунок А.6 – Аналіз та синтез і компілювання проекту

## Моделювання схеми логічного пристрою

Середовище Quartus передбачає можливість використання ряду професійних засобів моделювання. В рамках даного курсу лабораторних робіт їхня функціональність не є необхідною, тому для спрощення обмежимось роботою лише з вбудованим інструментом моделювання Quartus, який і розглянемо нижче.

### Створення тестових сигналів для моделювання

Перед початком моделювання необхідно створити тестові сигнали, що подаватимуться на схему. Для цього в Quartus передбачено спеціальний вид файлів *.vwf – векторні часові діаграми, створення та змінення яких виконується в окремому редакторі. Щоб створити часову діаграму необхідно зайти в головне меню File → New та у вікні, що відкриється та обрати Vector Waveform File, а у нових версіях Quartus University Program VWF (рис. А.7).

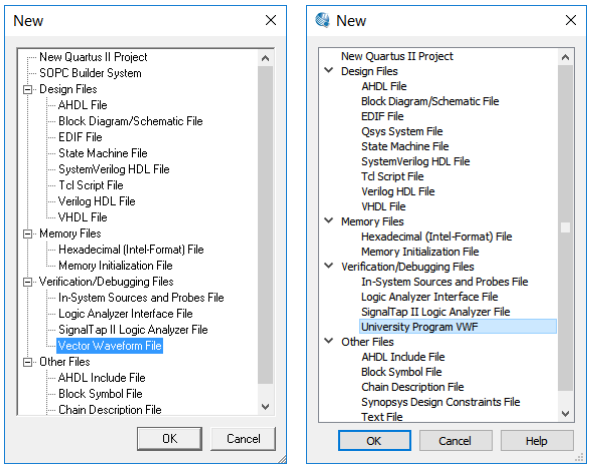



Рисунок А.7 – Створення файлу тестових сигналів в старій (зліва) та новій (справа) версіях Quartus

При цьому відкриється редактор, що має вигляд показаний на рис. А.8.

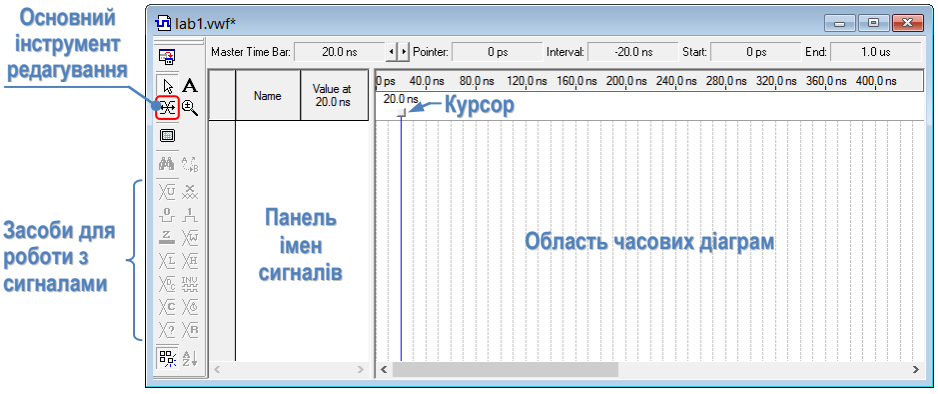



Рисунок А.8 – Редактор векторних часових діаграм (стара версія Quartus)

Редактори сигналів у старій і новій версіях Quartus мають незначні відмінності, зокрема у новій версії відсутній основний інструмент редагування сигналів, а панель інструментів розташована горизонтально. В усьому іншому вони є подібними, а робота в них здійснюється аналогічно.

Щоб додати сигнали до редактора необхідно правою кнопкою миші викликати контекстне меню у полі імен сигналів та обрати пункт Insert Node or Bus… (рис. А.9).

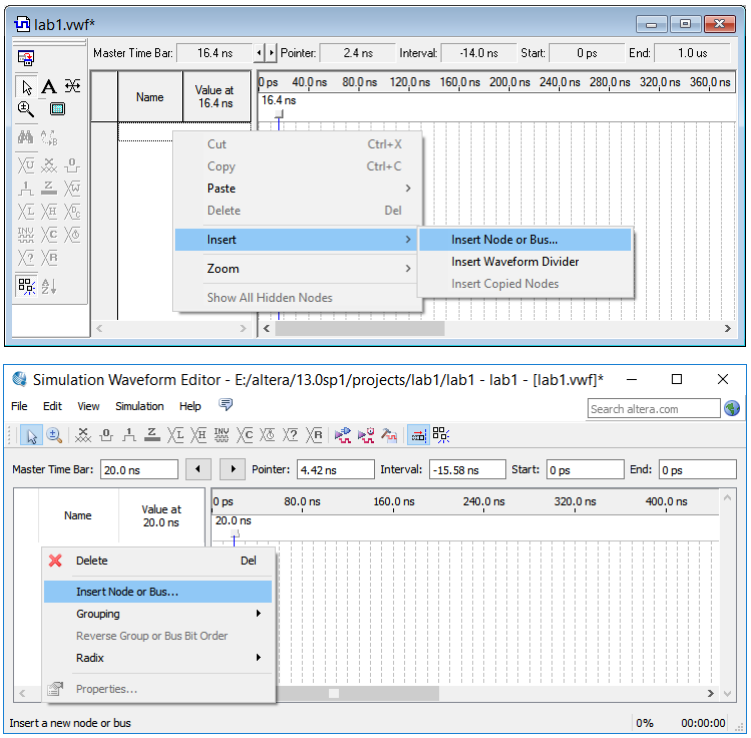



Рисунок А.9 – Додання сигналів у редакторі у старій (зверху) і новій (знизу) версіях Quartus

Процедура додавання сигналів показана на рисунку А.10 (послідовність дій, яку необхідно виконати у вікні Node Finder пронумеровано). *Зауваження*: на момент додання сигналів обов'язково має бути виконаний аналіз та синтез проекту або його компілювання, інакше сигнали у списку *не відображатимуться*.

Коли сигнали додані до редактора їх можна змінювати за допомогою відповідних засобів. Приклад формування сигналів для моделювання наведено на рис. А.11.

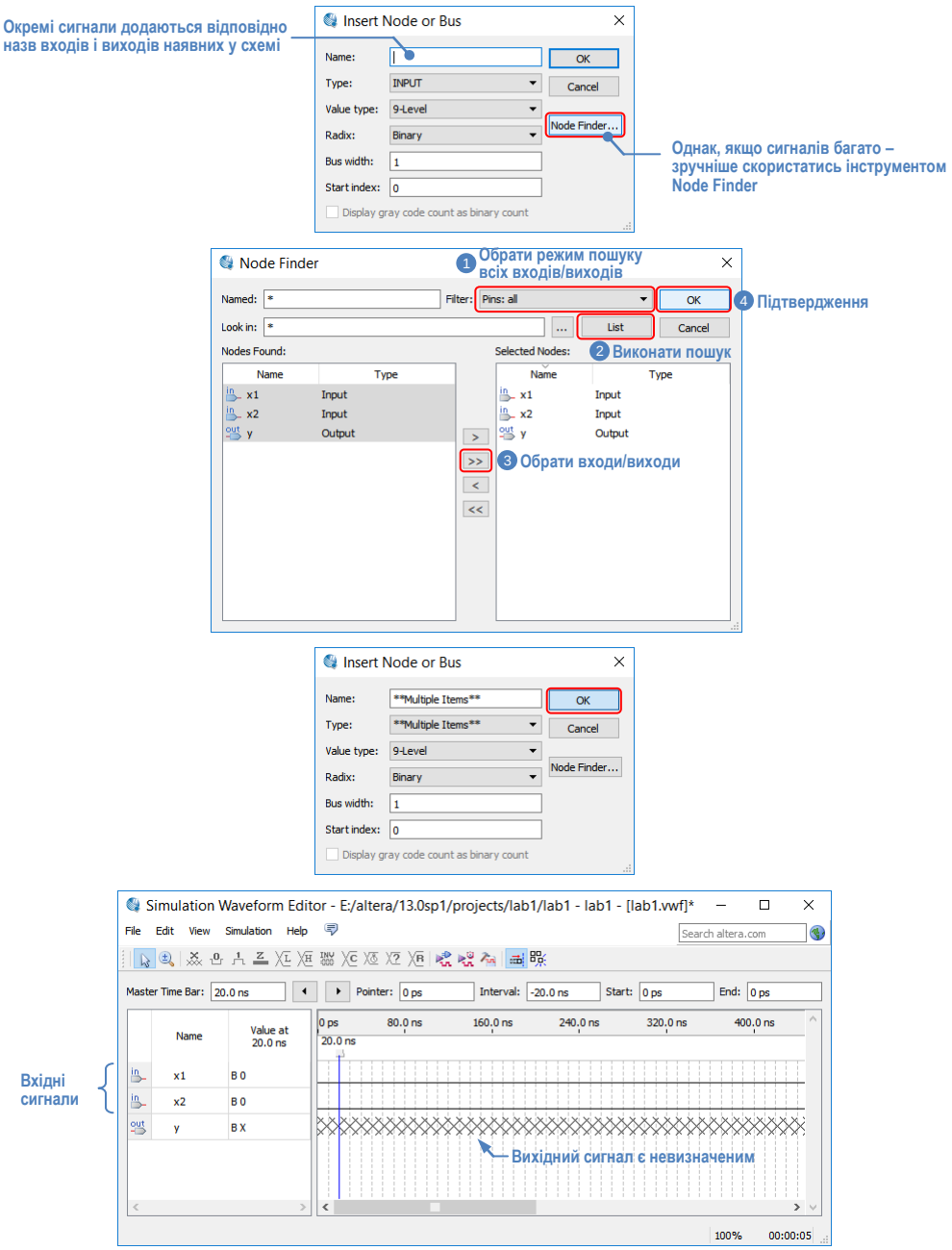



Рисунок А.10 – Додання сигналів до редактора

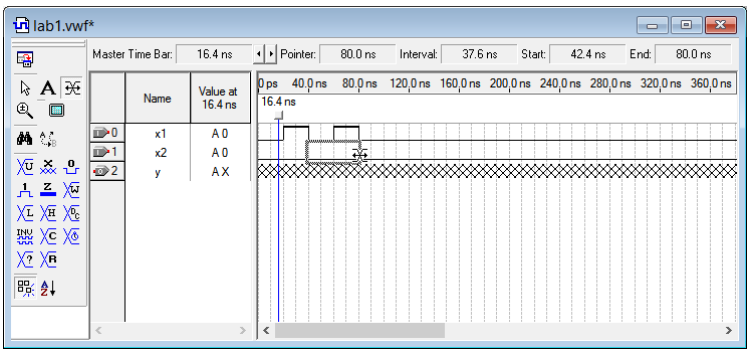



Рисунок А.11 – Формування сигналів у редакторі

### Моделювання логічних схем у нових версіях Quartus II

Запуск моделювання схеми в нових версіях Quartus виконується одразу в редакторі сигналів, для цього в ньому передбачені відповідні кнопки показані на рис. А.12.

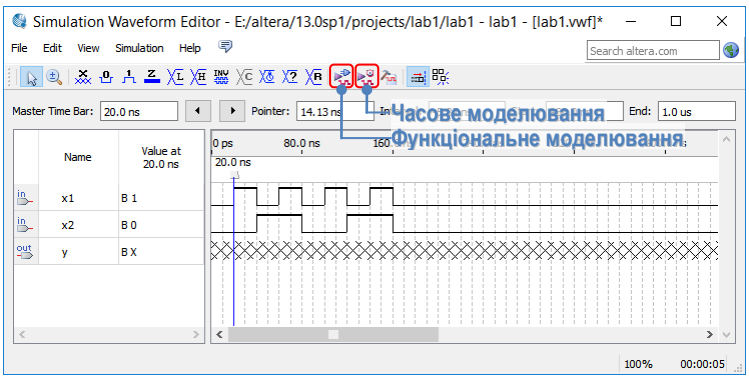



Рисунок А.12 – Режими моделювання в новому редакторі сигналів Quartus

Моделювання в часовому режимі як і в старіших версіях Quartus потребує компілювання проекту (зокрема, запуску TimeQuest Timing Analyzer). Часові діаграми з результатами моделювання відображаються в окремому вікні. Для схеми з рисунку А.5, результати моделювання в функціональному та часовому режимах наведено на рис. А.13 та А.14 відповідно.

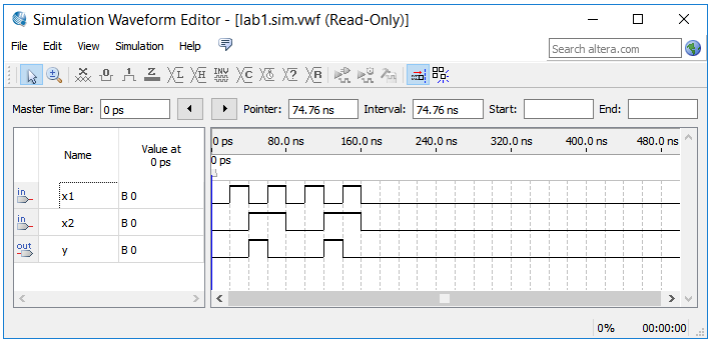



Рисунок А.13 – Моделювання в функціональному режимі

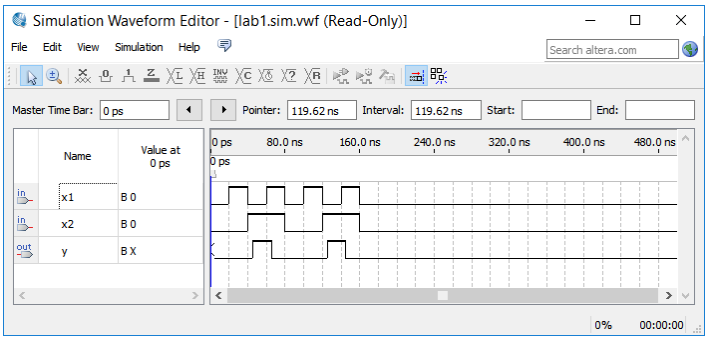



Рисунок А.14 – Моделювання в часовому режимі для мікросхеми Cyclon II

Якщо під час моделювання виникають помилки через некоректне встановлення середовища симуляції ModelSim або якщо ModelSim взагалі не був встановлений, можна скористатись старим вбудованим симулятором Quartus. Його можна обрати через головне меню Simulation → Options.

### Моделювання логічних схем у старих версіях Quartus II

Моделювання у Quartus до версії 10 виконується за допомогою спеціального інструменту доступного у головному меню Processing → Simulator Tool (рис. А.15).

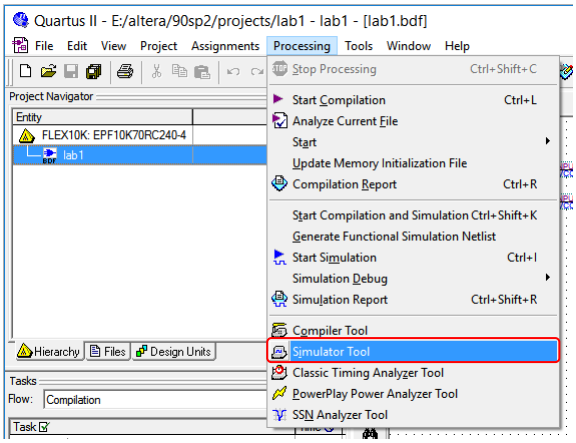



Рисунок А.15 – Виклик інструменту симуляції через головне меню

Сам інструмент моделювання налаштовується таким чином, як показано на рисунку А.16. Моделювання в даному випадку проводиться в функціональному режимі. Цей режим передбачає оцінювання роботи схеми без урахування затримок розповсюдження сигналів через логічні елементи.

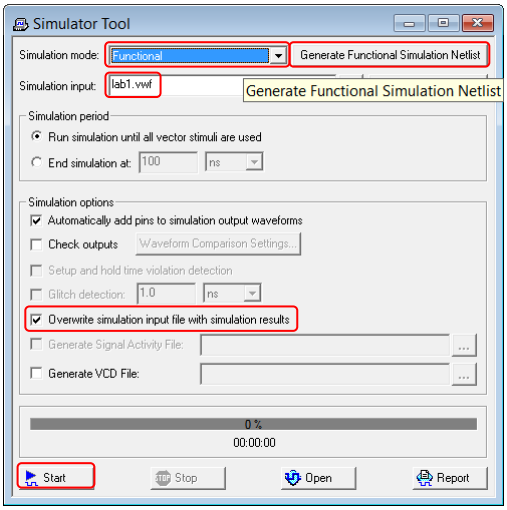



Рисунок А.16 – Налаштування інструменту моделювання

Функціональний режим моделювання потребує генерування спеціального списку з'єднань. Для цього перед моделюванням слід натиснути на кнопку Generate Functional Simulation Netlist. У якості вхідних даних для моделювання обираємо щойно створений файл часових діаграм. Зазвичай файл часових діаграм обирається автоматично, але якщо з якихось причин цього не відбулося або необхідно вказати іншу часову діаграму, то її можна змінити у полі Simulation Input. Крім того у вікні налаштувань доцільно ввімкнути перезапис вхідних даних результатами моделювання (поставити галочку в полі Overwrite simulation input file with simulation results). Це дозволяє відобразити результат моделювання одразу у вхідну часову діаграму (vwf-файл). Запуск моделювання виконується натисканням кнопки Start. Результати моделювання матимуть вигляд показаний на рис. А.17.

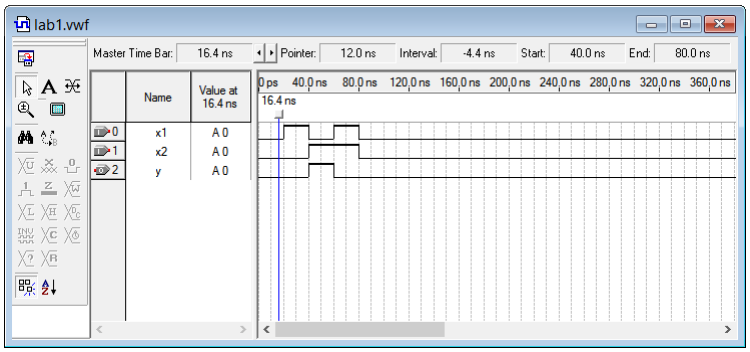



Рисунок А.17 – Приклад функціонального моделювання

Для проведення моделювання з урахуванням реальних затримок при розповсюдженні сигналів на конкретній ПЛІС, обирається режим часової (Timing) симуляції як показано на рис. А.18. *Зауваження*: моделювання в часовому режимі потребує обов'язкового компілювання проекту.

Результат моделювання схеми з рис. А.5 в часовому режимі для мікросхеми Flex показано на рис. А.19. Зверніть увагу на те, як змістився вихідний сигнал *y* та на появу *викиду* поблизу часової відмітки 100 нс.

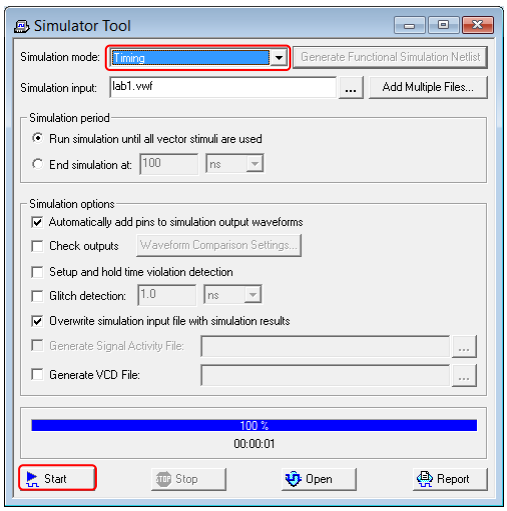



Рисунок А.18 – Налаштування інструменту моделювання для врахування затримок розповсюдження сигналів у схемі

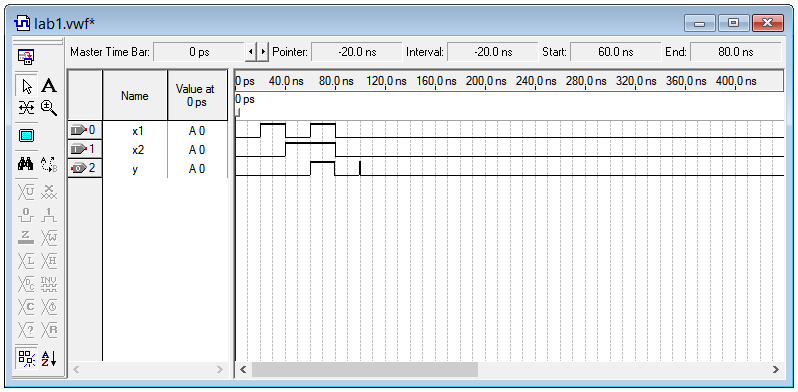



Рисунок А.19 – Моделювання з урахуванням затримок

## Прив’язування входів і виходів схеми конкретним виводам ПЛІС

При реалізації логічної схеми у вигляді фізичного пристрою на ПЛІС, на неї необхідно мати можливість подавати зовнішні сигнали. Щоб це здійснити, входи та виходи схеми повинні бути підключені до конкретних виводів ПЛІС. Для реалізації даної можливості в середовищі Quartus передбачено спеціальний інструмент, що називається Pin Planer. Його виклик можна здійснити за допомогою кнопки показаній на рис. А.20 або через головне меню Assigments → Pin Planer (комбінація клавіш Ctrl + Shift + N).

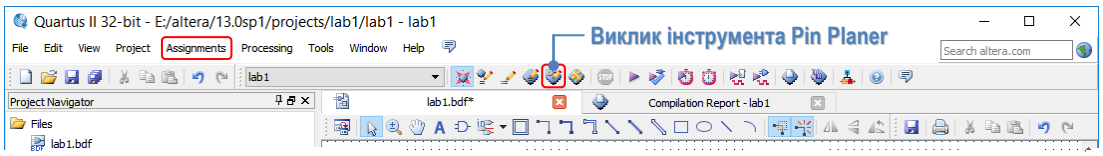



Рисунок А.20 – Виклик інструмента Pin Planer

Вікно інструмента Pin Planer показано на рисунку А.21.

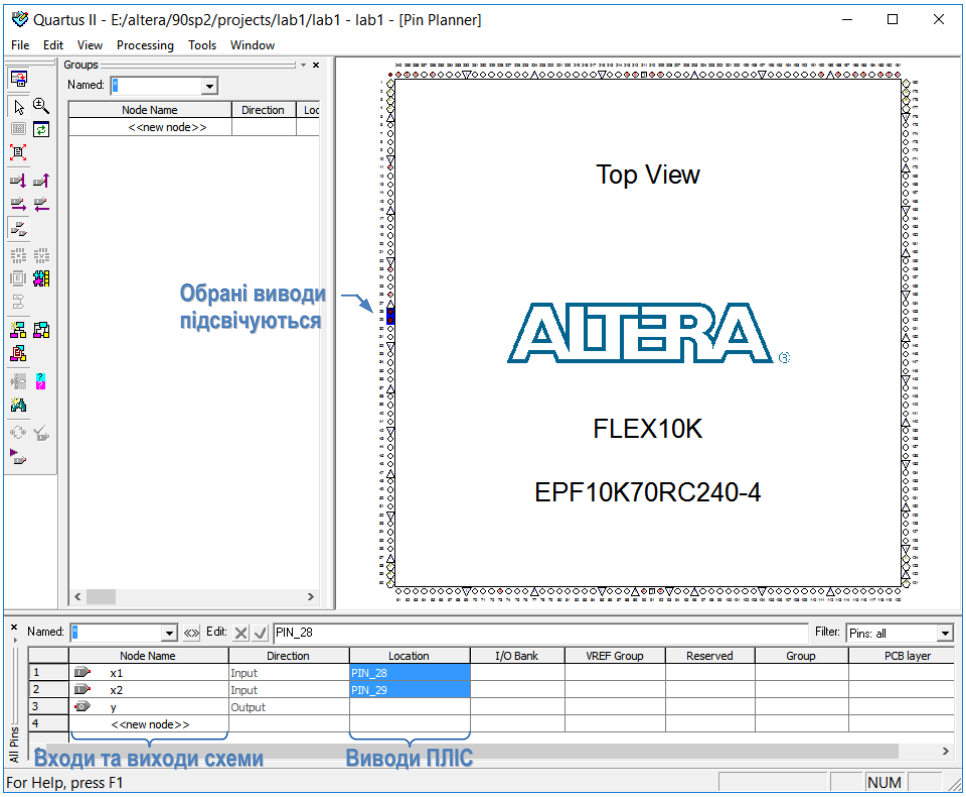



Рисунок А.21 – Вікно Pin Planer

У вікні інструмента Pin Planer кожному входу та виходу схеми, які перелічені у стовпці Node Name необхідно поставити у відповідність вивід ПЛІС у стовпці Location. Вибір виводів виконується згідно того, як на конкретній платі налагодження виконано підключення засобів введення-виведення інформації. Зокрема, на рис. А.21 входи схеми $x_1$ та $x_2$ підключаються до виводів 28 та 29 ПЛІС, які на платі UP2 відповідають двом кнопкам (див. додаток Б, табл. Б.1). Вихід схеми *y* можна підключити до будь-якого з сегментів семисегментного індикатора UP2. Наприклад, якщо для виводу обрати десяткову точку першого знакомісця, то в полі для сигналу *y* слід було б обрати вивід 14 (див. додаток Б, табл. Б.3).

Більш детально компоненти введення-виведення, що наявні на платах налагодження описані у додатку Б.

### Імпорт присвоювань

Присвоєні виводам ПЛІС назви можуть бути збережені в окремому файлі, а потім завантажені у необхідний момент до проекту. Для цього в

Quartus є спеціальний засіб, що називається Assignments Editor, який можна викликати через головне меню Assignments → Assignments Editor (комбінація клавіш Ctrl + Shift + A). Завантаження та збереження присвоєних виводам назв може виконуватись окремо через головне меню Assignments → Import Assignments та Assignments → Export Assignments відповідно.

При роботі з платою налагодження DE2, назви присвоєні виводами ПЛІС збережені в спеціальному файлі DE2_pin_assignments.csv (додається окремо). Щоб скористатись присвоюваннями їх потрібно імпортувати (Assignments → Import Assignments), а потім у схемі змінити назви виходів та входів на назви виводів, до яких підключені необхідні периферійні пристрої, засоби відображення та введення. При цьому, щоб зрозуміти які назви надані виводам можна викликати Assignments Editor або Pin Planer (вигляд вікна Pin Planer після імпорту DE2_pin_assignments.csv наведено на рис. А.22). Як видно, імена надані виводам ПЛІС мають більш-менш зрозумілий формат і відповідають назвам компонентів на платі DE2.

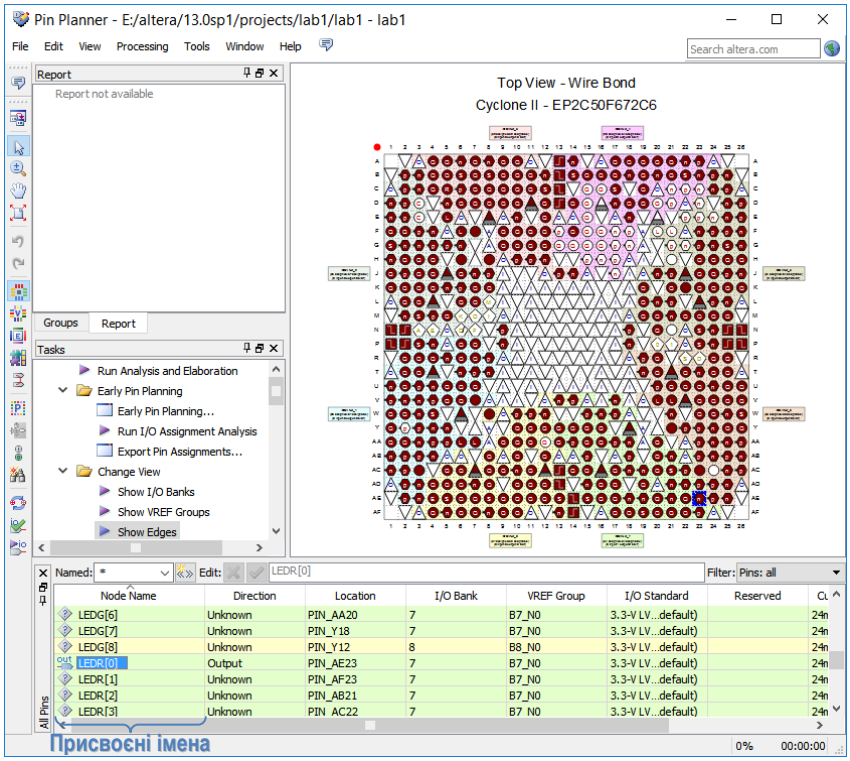



Рисунок А.22 – Вигляд вікна Pin Planer після імпорту присвоювань

Після зміни назв виводів та повного компілювання проекту (Ctrl + K) схема набуде вигляду показаного на рис. А.23. У прямокутниках під входами

та виходами схеми з’явилася інформація, до яких виводів ПЛІС вони підключені.

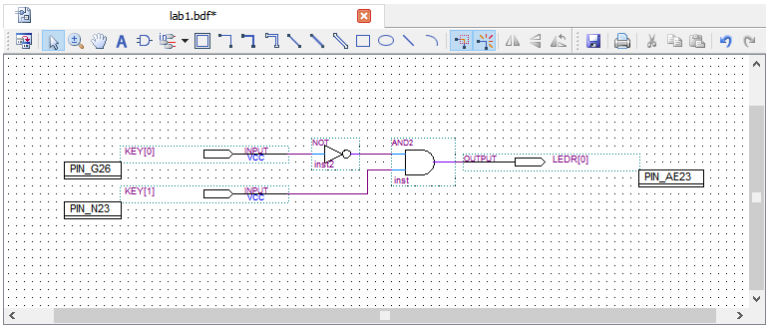



Рисунок А.23 – Схема після імпорту присвоювань та зміни імен її входів та виходів

## Програмування ПЛІС

Програмування ПЛІС у Quartus виконується за допомогою окремого інструменту, що так і називається Programmer. Перш ніж ним скористатись може знадобитись встановлення драйверів. Зокрема, це буде необхідно, якщо пристрій програмування підключається до комп’ютера через шину USB. Драйвер USB програматора (USB-Blaster) знаходиться у папці *drivers* каталогу, в який було встановлено Quartus II (наприклад, на Windows це може бути C:\altera\13.0sp1\quartus\drivers). У випадку, якщо застосовується LPT програматор (Byte-Blaster), встановлення драйверів не потрібне.

Виклик інструмента програмування здійснюється через головне меню Tools → Programmer або натисканням на кнопку показану на рис. А.24.

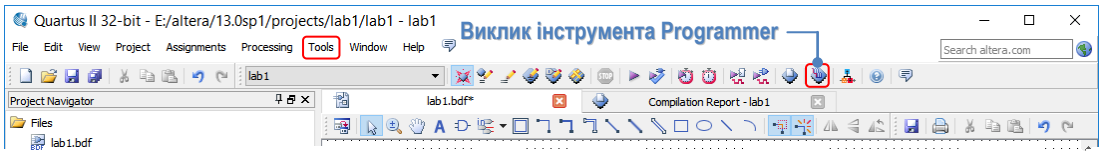



Рисунок А.24 – Виклик інструмента для програмування ПЛІС

Вікно інструмента Programmer показано на рис. А.25. У його вікні відображаються файл прошивки для ПЛІС і в нових версіях Quartus мікросхема, що програмуватиметься. Файл прошивки (sof-файл) генерується при виконанні повного компілювання проекту (див. рис. А.6). У інструменті програмування, як правило, файл прошивки (якщо він був створений під час

компілювання) з’являється автоматично. Якщо цього не відбулось або необхідно запрограмувати інший файл, прошивки можна додавати та змінювати вручну кнопками Add File, Delete та Change File.

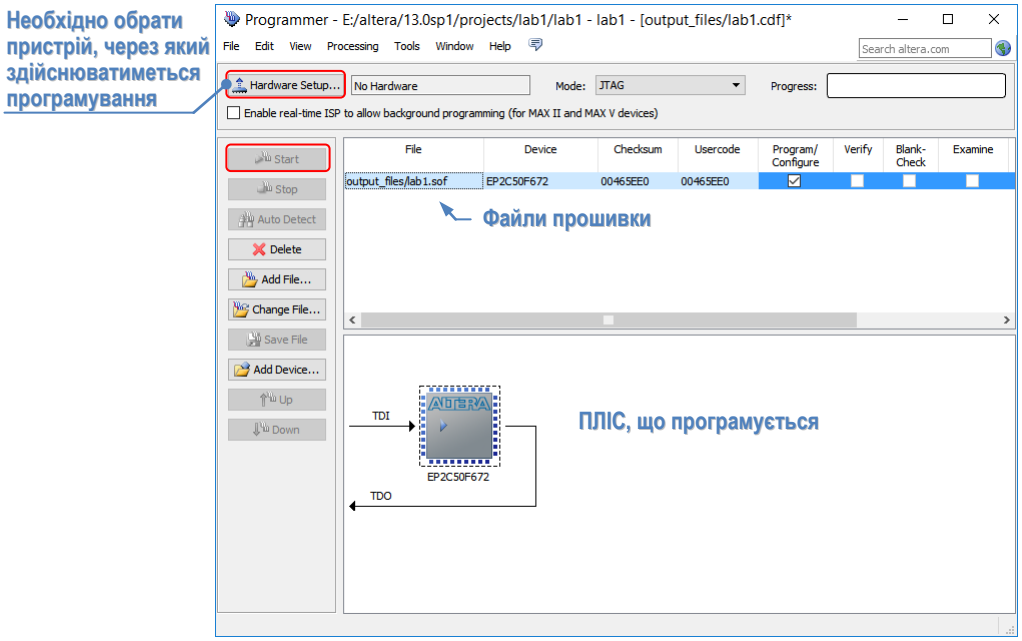



Рисунок А.25 – Вікно інструмента програмування Programmer

Щоб завантажити прошивку до мікросхеми ПЛІС, необхідно обрати коректний засіб програмування. Це робиться натисканням на кнопку Hardware Setup… та вибором у вікні, що відкриється необхідного програматора у полі Currently Selected Hardware. Після цього вікно Hardware Setup можна закрити і якщо все було зроблено вірно, то у вікні Programmer кнопка початку програмування Start має стати активною. Натиснувши на кнопку Start до ПЛІС буде завантажено створену прошивку.

***Зауваження***: при програмуванні ПЛІС на платі DE2 USB-кабель необхідно підключати до роз’єму USB Blaster Port (див. рис. Б.3)

# ДОДАТОК Б
# ЗАСОБИ ВВЕДЕННЯ-ВИВЕДЕННЯ ПЛАТ НАЛАГОДЖЕННЯ

Плати налагодження, з якими ми працюватимемо, мають значний набір периферійних пристроїв та засобів введення-виведення. Проте, не всі з них необхідні для виконання даного курсу лабораторних робіт. З огляду на це, обмежимось розглядом особливостей підключення до ПЛІС лише тих засобів введення-виведення, які нам реально знадобляться, а саме: кнопок, перемикачів, світлодіодів та семисегментних індикаторів.

## Плата налагодження UP2

Схематично плата налагодження UP2 показана на рис. Б.1.

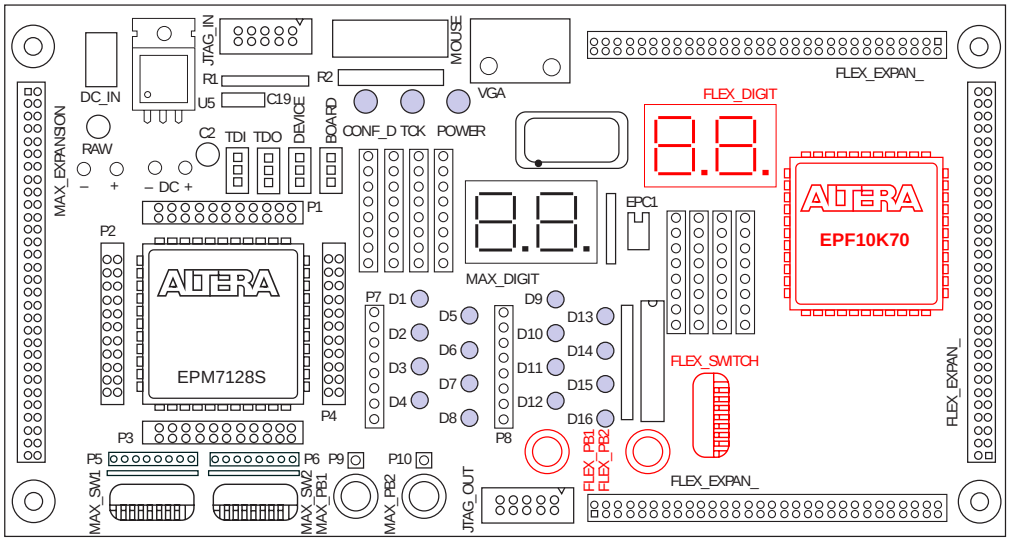



Рисунок Б.1 – Плата налагодження UP2:
червоним позначено компоненти, що розглядаються нижче

До ПЛІС EPF10K70 на платі налагодження UP2 підключено дві кнопки FLEX_PB1 та FLEX_PB2 (рис. Б.1, табл. Б.1). Кожна з них зі сторони ПЛІС підтягнута через 10 КОм резистор до живлення. Таким чином, коли кнопка відпущена на вході EPF10K70 формуватиметься логічна «1», а при натисканні на неї на вхід ПЛІС подаватиметься рівень логічного «0».

Таблиця Б.1 – Підключення кнопок до контактів
ПЛІС EPF10K70 на платі UP2

| Кнопка | Контакт ПЛІС |
|---|---|
| **FLEX_PB1** | PIN_28 |
| **FLEX_PB2** | PIN_29 |

До ПЛІС FLEX (EPF10K70) підключено вісім перемикачів, розміщених в єдиному корпусі FLEX_SWITCH (рис. Б.1, табл. Б.2). Дані перемикачі функціонують наступним чином: при розмиканні перемикача на вхід ПЛІС подається логічна «1», а при його замиканні – логічний «0».

Таблиця Б.2 – Підключення перемикачів до контактів ПЛІС EPF10K70 на платі UP2

| Перемикач | Контакт ПЛІС |
|---|---|
| **FLEX_SWITCH-1** | PIN_41 |
| **FLEX_SWITCH-2** | PIN_40 |
| **FLEX_SWITCH-3** | PIN_39 |
| **FLEX_SWITCH-4** | PIN_38 |
| **FLEX_SWITCH-5** | PIN_36 |
| **FLEX_SWITCH-6** | PIN_35 |
| **FLEX_SWITCH-7** | PIN_34 |
| **FLEX_SWITCH-8** | PIN_33 |

До ПЛІС EPF10K70 на платі налагодження платі UP2 безпосередньо підключено подвійний семисегментний індикатор FLEX_DIGIT (рис. Б.1). У табл. Б.3 наводиться відповідність підключення кожного із його сегментів до контактів мікросхеми EPF10K70. Найменування сегментів індикатора показано на рис. Б.2. Ввімкнення сегментів виконується низьким рівнем (логічним 0).

Таблиця Б.3 – Підключення семисегментних індикаторів до контактів ПЛІС EPF10K70 на платі UP2

| Сегмент | Контакт ПЛІС, підключений до першого знакомісця | Контакт ПЛІС, підключений до другого знакомісця |
|---|---|---|
| **a** | PIN_6 | PIN_17 |
| **b** | PIN_7 | PIN_18 |
| **c** | PIN_8 | PIN_19 |
| **d** | PIN_9 | PIN_20 |
| **r** | PIN_11 | PIN_21 |
| **f** | PIN_12 | PIN_23 |
| **g** | PIN_13 | PIN_24 |
| **dp** | PIN_14 | PIN_25 |

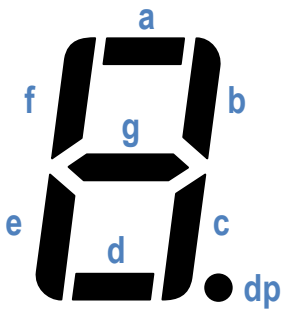



Рисунок Б.2 – Позначення сегментів для одного знакомісця семисегментного індикатора

## Плата налагодження DE2

Плата налагодження DE2 показана на рис. Б.3.

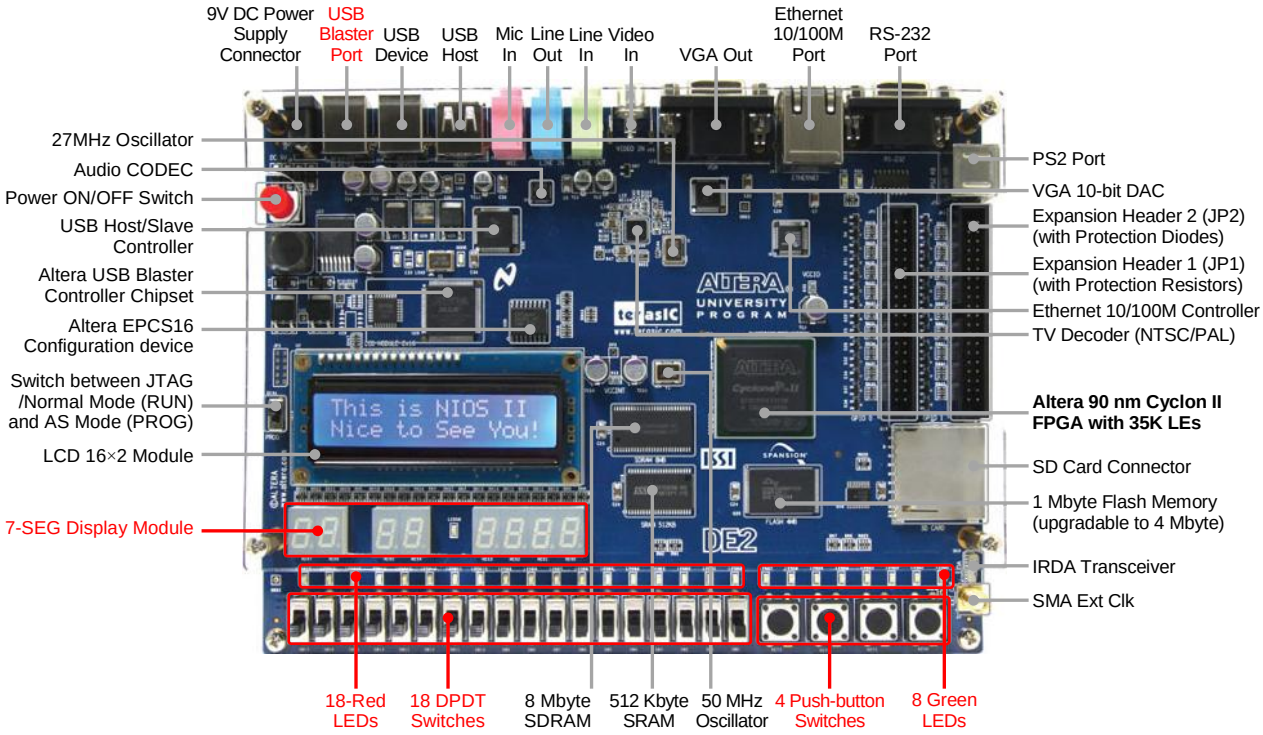
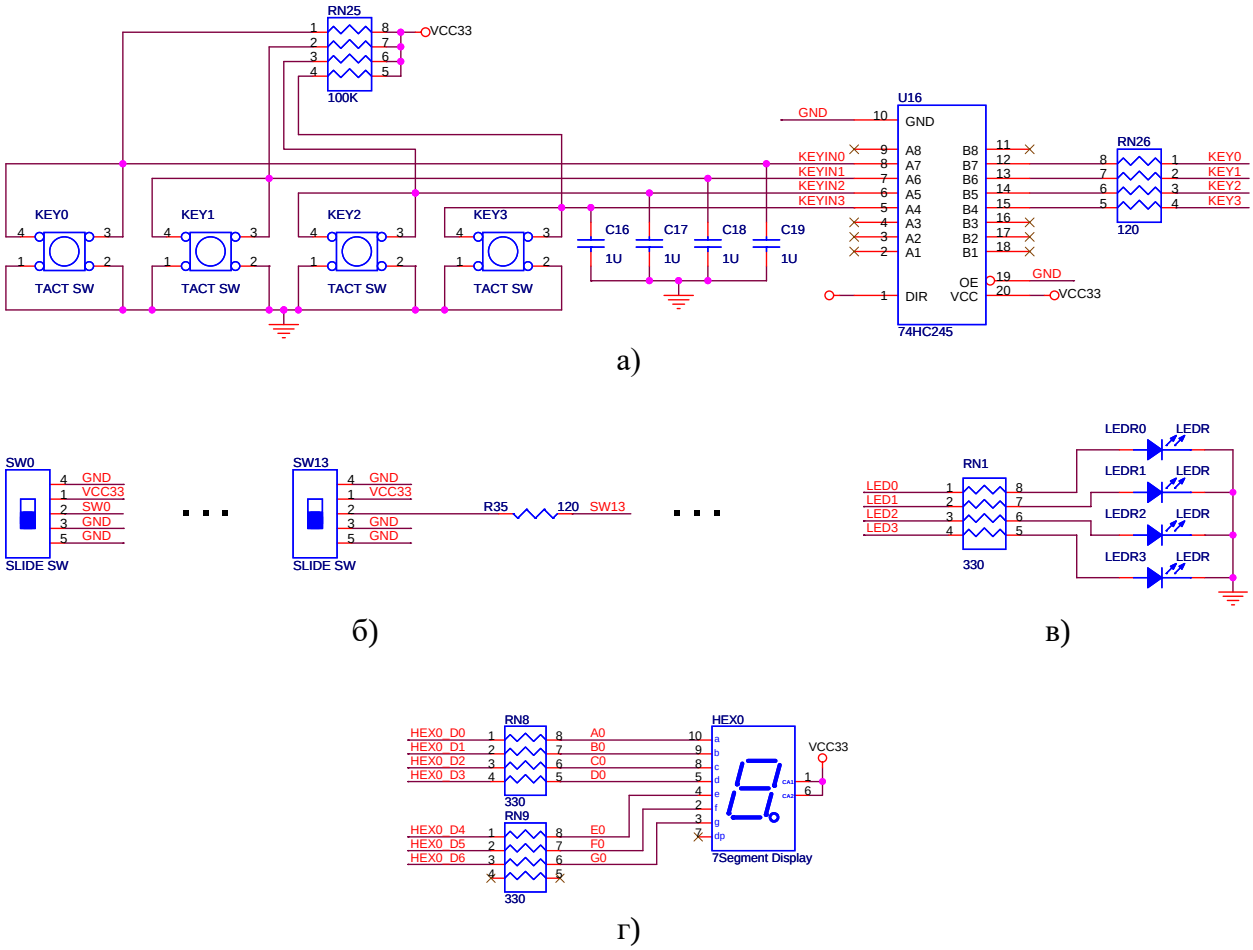



Рисунок Б.3 – Плата налагодження DE2:
червоним позначено компоненти, що використовуються
в рамках даного курсу лабораторних робіт

Підключення периферійних пристроїв та засобів введення-виведення до ПЛІС EP2C35 на платі налагодження DE2 виконано згідно таблиці, що наводиться в окремому файлі DE2_pin_assignments.csv. Даний файл можна одразу імпортувати до середовища Quartus. Як це зробити описано в додатку А, підрозділі «Імпорт присвоювань».

Рівні сигналів, які виникають при натисканні на кнопки та перемикачі, а також якими засвічуються світлодіоди та сегменти семисегментних індикаторів, можна визначити за схемою плати DE2. Тому наведемо основні способи підключення периферії до ПЛІС Cyclon II (EP2C35), що використані на платі DE2 (див. рис. Б.4).




а)

б) в)

г)

Рисунок Б.4 – Схеми підключення елементів введення-виведення на платі DE2:

а) кнопок;

б) перемикачів;

в) світлодіодів;

г) семисегментних індикаторів

Більше документації до плати DE2 можна завантажити за посиланням:

https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=53&No=30&PartNo=4

## СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ


1. Варфоломєєв А.Ю. Функціонально-логічне проектування: Комбінаційні пристрої [Електронний ресурс] / КПІ ім. Ігоря Сікорського. – Електронні текстові данні (1 файл). – Київ : КПІ ім. Ігоря Сікорського, 2017 р. – 135 с.
2. Пухальский Г.И. Цифровые устройства: Учебное пособие для втузов / Пухальский Г.И., Новосельцева Т.Я. – СПб.: Политехника, 1996. – 885 с.
3. Харрис Д.М. Цифровая схемотехника и архитектура компьютера / Д.М. Харрис, С.Л. Харрис. – Нью-Йорк: Elsevier, 2013. – 1621 с.
4. Шило В.Л. Популярные цифровые микросхемы: Справочник. – М.: Радио и связь, 1987.
5. Угрюмов Е.П. Цифровая схемотехника: Учеб. пособие для вузов. – 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2004.
6. Потёмкин И.С. Функциональные узлы цифровой автоматики. – М.: Энергоатомиздат, 1988.